**Objectives**

1. Research information about software for a specific operating system (OS) environment. You will be assigned one of the operating systems form the list below. You will also be provided with a list of topics to investigate.

1. Organize your rough research information into a list of topics, sub-topics and facts. This process will involve identifying sub-topics, rearranging your rough research notes, and selecting (or highlighting) interesting facts.
2. Report a summary of your research in the form of a “concept map”. Use the PowerPoint template provided as a starting point. The concept map should only include the best and most interesting information from your organized research notes.
3. Your concept map can be created using: Smart Ideas, Prezi, PowerPoint or other similar applications.

****

**Step 1 – Organized Research**

Research information about your assigned operating system (OS) environment.

* Guide your research according to the suggested topic list below
* Feel free to copy-and-paste as long as you keep track of your bibliographic references.
* Do not be too picky or concerned about formatting as you will organize this information later in step 2
* Select things that look interesting and don’t forget to include graphics images as well
* Upload your rough research notes to your repository when you are done.

Topic A – Application Software

Provide a summary of most important user application software targeted by this operating system and how it is similar to and deferent from standard PC software. Suggested sub-topics include:

* User (client) or network (server) applications
* Batch (run without user input) or interactive (user focused) processing
* Off-the-shelf (purchased) or custom developed applications
* Programming environment and languages supported

QNX is a mobile operating system that was originally developed for embedded systems. The operating system’s developer, QNX Software Systems, was acquired by Research in Motion (RIM) and the OS adapted for use in the BlackBerry Playbook tablet. The version of QNX used in the Playbook is known as the QNX Neutrino real-time operating system (RTOS). The Unix-like QNX OS is microkernel-based, which means that it is a small program and elements of the operating system run as tasks that the developer can turn off if they aren’t required in a particular system. This contrasts with the traditional monolithic OS architecture, in which the operating system runs as a single program. QNX competes in the tablet market with Apple’s iOS and Google’s Android operating systems. The operating system is also expected to replace the BlackBerry OS used in the company’s smartphones. The QNX® SDK for Apps and Media leverages open technologies for application development (HTML5, Qt, OpenGL ES) and allows device manufacturers to build compelling mobile-like interfaces with full multimedia capabilities, powered by secure, reliable, and field-proven QNX technologies.

Topic B – Hardware

Provide a summary of the hardware targeted by this operating system and how it is similar to and deferent from standard PC hardware. Suggested sub-topics include:

* Speed of processors / memory
* Capacity of memory / attached disks
* Is it designed for home / office / corporate data center / industrial use
* Is it designed for client / server / network use

BlackBerry® QNX, with support from hardware and silicon partners, offers a broad and highly optimized level of hardware support for its software, including QNX® Software Development Platform 7.0 (QNX® SDP 7.0). QNX SDP 7.0 provides a 32-bit and 64-bit OS that builds on the proven reliability of BlackBerry QNX technology, and raises the bar for security and performance in mission critical applications. The QNX SDP 7.0 BSPs support a wide array of boards based on ARMv7, ARMv8, and x86 architectures. Source code is available for all QNX BSPs to allow modifications for your custom boards.

Topic C – User Interface

Provide a summary of the user interface and input devices targeted by this operating system and how it is similar to and deferent from a standard PC. Suggested sub-topics include:

* Does it support a windowed environment, command line, or network users
* Does it support multiple users at a time or single users
* Does it support multiple applications or a single application at a time
* Does it get rebooted (powered on / off) or is it always on

**Best-in-class support for HTML5**

The QNX HTML5 engine provides support for the HTML5 standard, and related standards and technologies such as CSS3, the JavaScript scripting language, plus associated standards, such as AJAX, JavaScript Object Notation (JSON), and XML. It also supports HTML5-specific features such as WebSocket, WebGL, session storage, offline applications, worker threads, DOM improvements, and the <canvas>, <audio>, and <video> elements. HTML5 applications can support a variety of user interface technologies, including interactive displays, audio, and video. With HTML5, developers can use a common tool set to build applications for QNX-based embedded devices, mobile devices, or applications to be hosted in the cloud.

**Fast time-to-market**

Using HTML5 or Qt to design the system’s user interface can dramatically reduce development efforts, especially when compared to designing with traditional embedded UI toolkits. With the QNX SDK for Apps and Media, developers have a common tool set to build, style, and animate applications for embedded devices. Pre-integrated software with support for HTML5 and Qt 5.3 will move you to the prototyping phase of your next project, fast.

**Simplified system design**

The application framework and platform services allow development teams to seamlessly combine multiple UI technologies onto a common UI. The QNX SDK for Apps and Media includes a fully ported, integrated, and optimized version of Qt that enables development teams to "Code Less. Create More".

**Reliability and Security**

The QNX SDK for Apps and Media provides a partitioned user interface and sandbox for ultimate reliability. Developers can seamlessly blend HTML5, OpenGL ES, and Qt applications with one of several supported partner HMI toolkits. With this approach, applications can be isolated from each other to ensure incorrect behavior of one application can’t affect other applications or bring down the system.

**Rebooting**

You rarely need to reboot a QNX Neutrino system. If a driver or other system process crashes, you can usually restart that one process. ... To shut down or reboot the system, use the shutdown command. You can do this only if you're logged in as root.

**Single Platform**

Due to the inherent connectivity of the QNX SDK for Apps & Media, a single platform supports media sharing among multiple users, immediate and unique identification of media devices and streams (including mobile phones and media players), auto synchronization to databases, and multiple playback and record paths.

**Multiple Users**

QNX Neutrino is a multiuser operating system; it lets multiple users log in and use the system simultaneously, and it protects them from each other through a system of resource ownership and permissions. Depending on the configuration, your system boots into text mode and prompts you for your user ID and password.

Topic D – Device Management

Provide a summary of the devices (disks, printers, etc.) and memory managed by this operating system and how it is similar to and deferent from a standard PC. Suggested sub-topics include:

* What types of disk drives and file systems does it support
* What type of input devices does it support
* What type of output devices does it support

**CD-ROMs and DVDs**

You usually attach CD and DVD drives to a SCSI or EIDE(ATA) bus; which driver you use depends on the bus. Ensure that the hardware is set up correctly and that the BIOS detects the hardware properly. If you attached the drive to an EIDE bus, simply use the devb-eide driver. If the drive is on a SCSI bus, you need to determine the proper driver for your SCSI interface

**Floppy disks**

The driver for a floppy drive is devb-fdc. In order to use a floppy disk, you need to ensure that the floppy controller is enabled in the BIOS, and that the BIOS is configured to recognize the correct type of floppy drive (e.g. 1.44MB/2.88MB). The driver uses these locations as default:

* I/O port 0x3f0
* IRQ 6
* DMA 2

If your controller is located at a different address, you can change these locations in the driver's options.

**Hard disks**

A self-hosted system, by default, detects the disk controller that's installed on the system, and then starts the appropriate driver for it. On a self-hosted system, the diskboot utility in the OS image starts the block I/O drivers. If you want to change the way that the driver is started, you'll need to change the startup image and the options to diskboot.

**EIDE**

EIDE interfaces use the devb-eide driver, which by default automatically detects the interface and devices attached to it. This driver includes support for UDMA (Ultra Direct Memory Access) modes, along with the generic PIO (Programmed Input/Output) modes. The supported hardware list includes adapters and their supported features; see the introduction to this chapter. You can start the devb-eide driver without any options and, by default, it automatically detects the EIDE controller on the system: devb-eide &

**SCSI devices**

A SCSI (Small Computer Systems Interface) bus is simply another bus that you can attach multiple peripherals to. Neutrino supports many brands and varieties of SCSI adapters; see the devb-\* (block-oriented) drivers in the Utilities Reference. When the SCSI driver starts up, it scans the bus for attached devices. When the driver finds a supported device, it creates an entry in the /dev directory (e.g. a hard drive is hdx, where x is the number of the drive, starting from 0).

**SCSI RAID**

Currently, Neutrino supports only hardware RAID (Redundant Arrays of Independent Disks) devices. There are many third-party solutions for SCSI RAID available for Neutrino; search for them on the Internet.

**LS-120**

LS-120 is a SuperDisk drive that uses new technology to greatly improve head alignment, enabling a much greater storage capacity (120 MB) than conventional 3.5-inch disks. Neutrino treats an LS-120 drive like an EIDE drive.

**ORB**

An ORB drive is a fast, large-capacity, removable storage disk drive that uses 3.5″ storage media and attaches to the EIDE (ATA) chain. Ensure that the hardware is set up correctly and that the BIOS detects the hardware properly. An ORB drive is simple to set up, and appears in the /dev directory as a hard disk. For example: The hard disk as a primary master appears as /dev/hd0. The ORB drive set up as a primary slave appears as /dev/hd1. To mount an ORB drive: mount /dev/hd1 /fs/orb\_drive.

**Zip and Jaz disks**

Zip and Jaz disks are large-capacity removable storage disks, used for backing up hard disks and for transporting large files. These disks attach to the EIDE(ATA) chain. Before you attempt to use them, ensure that the hardware is set up correctly and that the BIOS detects the hardware properly. These drives are simple to set up, and they appear in the /dev directory as a hard disk. For example: The hard disk set up as a primary master appears as /dev/hd0. The Zip disk set up as a primary slave appears as /dev/hd1.To mount the drive, type: mount /dev/hd1 /fs/zip\_drive

**Magnetic optical drives**

Magnetic optical (MO) drives are usually attached to a SCSI or EIDE (ATA) bus. Before you attempt to use the drive, ensure that the hardware is set up correctly and that the BIOS detects the hardware properly. The driver that you need depends on whether the drive is attached to a SCSI or EIDE interface. If it's SCSI, you'll need to determine the proper driver for your SCSI interface. If it's EIDE, simply use the devb-eide driver. For more information, see “Hard disks,” above. The drivers for optical disks load the cam-optical.so shared object, which provides a common access method for optical disks. The MO drive should appear in your /dev directory as /dev/mox, where x is the number of the drive, starting at 0. To mount the drive, type: mount /dev/mo0 /fs/mo\_drive

**RAM disks**

A RAM disk is a storage area that exists only in memory but looks like a hard disk. You can add one to your system by using devb-ram, but this is a RAM disk with the overhead of a block filesystem; by default, it's initialized and formatted for an fs-qnx4.so filesystem (unless you specify the ram nodinit option).

**Mice and keyboards**

Mice and keyboards both use the devi-hirun driver. The type of mouse attached to your system determines which options you need to use. For a serial mouse, you need to specify the correct protocol (e.g. the Microsoft Mouse protocol). Keyboards are detected on these interfaces: AT-style adapters appear as /dev/kbddev. PS/2 keyboards appear as /dev/kbd.

**Touchscreens**

Neutrino supports various touchscreens; check the list of supported hardware on our website to determine which driver to use for yours. See also the devi-\* input drivers in the Utilities Reference. Determine which options are appropriate for your setup, and then start the driver. For example, here's how to start the driver for a Dynapro SC4 touchscreen: devi-dyna dyna -4 fd -d/dev/ser1 &

**Audio cards**

By default, the operating system detects your audio card. The enumerators identify the card and use io-audio to start it. Audio drivers in Neutrino are very simple to initialize. When you use io-audio, you can use the -d option to pass the driver: io-audio -vv -d audiopc &

**ISA cards**

ISA cards are either Plug-and-Play or not. You typically have to manually set up non-PnP ISA devices. In order to identify your device, you need to have the manual for your device or have a way to contact your device's manufacturer (e.g. via their website). There isn't currently a Neutrino utility that lists the ISA devices that are installed on a system.

**PCI Cards**

The device enumerator should start PCI cards correctly. If your PCI card doesn't work, swap PCI slots. Sometimes the IRQ that's assigned to the particular slot doesn't work well with the card.

Topic E – Security

Provide a summary of the security features provided by this operating system and how it is similar to and deferent from a standard PC. Suggested sub-topics include:

* What types of user accounts and user permissions does it support
* How does it protect against conflicts / interference between legitimate application processes
* How does it protect against malicious software
* How does it support software updates and security updates

**DEFENCE**

Military-grade security and reliability in mission-critical government and military systems where information is vital and lives can be at stake, downtime is not an option.

The need for a highly reliable, secure, and fast operating system is crucial. It is these systems that require the highest level of confidence and fault tolerance that comes from a proven technology, battle-hardened in over 30 years of field deployment.

Thanks to the true microkernel architecture of the QNX Neutrino® RTOS, full memory protection is built in. Any component can fail and be dynamically restarted without corrupting the microkernel or other components. If a failure does occur, a QNX-based system has the capability for self-healing through critical process monitoring and customizable recovery mechanisms.

**VIRUSES**

The hosts for a virus are system-call interfaces that are accessible from the point of entry (an infected program), such as sendmail or an HTTP server. The hosts are platform-specific, so a virus for Linux would in all likelihood terminate the host under Neutrino as soon as it tried to do anything damaging.

The viruses that circulate via email are OS-specific, generally targeted at Windows, and can't harm Neutrino systems, since they simply aren't compatible. Most UNIX-style systems aren't susceptible to viruses since the ability to do (much) damage is limited by the host. We have never heard of a true virus that could infect Neutrino.

In addition, since deployed Neutrino systems are highly customized to their designated application, they often don't contain the software that's open to such attacks (e.g. logins, web browsers, email, Telnet and FTP servers).

**Neutrino security in general**

Neutrino is a UNIX-style operating system, so almost all of the general UNIX security information (whether generic, Linux, BSD, etc.) applies to Neutrino as well. A quick Internet search for UNIX or Linux security will yield plenty of papers. You'll also find many titles at a bookstore or library.

We don't market Neutrino as being either more or less secure than other operating systems in its class. That is, we don't attempt to gain a security certification such as is required for certain specialized applications. However, we do conduct internal security audits of vulnerable programs to correct potential exploits.

For flexibility and familiarity, Neutrino uses the generic UNIX security model of user accounts and file permissions, which is generally sufficient for all our customers. In the embedded space, it's fairly easy to lock down a system to any degree without compromising operation. The ultrasecure systems that need certifications are generally servers, as opposed to embedded devices.

Topic F – Network Connectivity

Provide a summary of the network connectivity provided by this operating system and how it is similar to and deferent from a standard PC. Suggested sub-topics include:

* Is the computer stand-alone or part of a larger network
* What type of network and internet connections does it provide
* Does it provide other services such as backup, firewall, etc.

**IO-PKT\***

The io-pkt\* component is the active executable within the network subsystem. Acting as a kind of packet redirector/multiplexer, io-pkt\* is responsible for loading protocol and driver modules based on the configuration given to it on its command line (or via the mount command after it's started).

Employing a zero-copy architecture, the io-pkt\* executable efficiently loads multiple networking protocols or drivers (e.g., lsm-qnet.so) on the fly— these modules are shared objects that install into io-pkt\*.

The io-pkt stack is very similar in architecture to other component subsystems inside of the Neutrino operating system. At the bottom layer, are drivers that provide the mechanism for passing data to and receiving data from the hardware. The drivers hook into a multi-threaded layer-2 component (that also provides fast forwarding and bridging capability) that ties them together and provides a unified interface for directing packets into the protocol-processing components of the stack. This includes, for example, handling individual IP and upper-layer protocols such as TCP and UDP.

In Neutrino, a resource manager forms a layer on top of the stack. The resource manager acts as the message-passing intermediary between the stack and user applications. It provides a standardized type of interface involving open(), read(), write(), and ioctl() that uses a message stream to communicate with networking applications. Networking applications written by the user link with the socket library. The socket library converts the message-passing interface exposed by the stack into a standard BSD-style socket layer API, which is the standard for most networking code today.

**Stand Alone**

QNX can work as a single, logical machine instead and merged so a number of QNX machines on a network. QNX can be scaled down to a small, stand-alone, ROM-based embedded system, or scaled up to a full X Window, TCP/IP, NFS, etc. equipped workstation OS, or scaled out to encompass several hundred nodes on a network acting as a single machine. QNX can also support Doom.

**WiFi Configuration Using WPA and WEP**

Wi-Fi capability is built into the two hc variants of the stack (io-pkt-v4-hc and io-pkt-v6-hc). The NetBSD stack includes its own separate 802.11 MAC layer that's independent of the driver. Many other implementations pull the 802.11 MAC inside the driver; as a result, every driver needs separate interfaces and configuration utilities. If you write a driver that conforms to the stack's 802.11 layer, you can use the same set of configuration and control utilities for all wireless drivers.

The networking Wi-Fi solution lets you join or host WLAN (Wireless LAN) networks based on IEEE 802.11 specifications. Using io-pkt, you can:

connect using a peer-to-peer mode called ad hoc mode, also referred to as Independent Basic Service Set (IBSS) configuration

either act as a client for a Wireless Access Point (WAP, also known as a base station) or configure Neutrino to act as a WAP. This second mode is referred to as infrastructure mode or BSS (Basic Service Set).

Ad hoc mode lets you create a wireless network quickly by allowing wireless nodes within range (for example, the wireless devices in a room) to communicate directly with each other without the need for a wireless access point. While being easy to construct, it may not be appropriate for a large number of nodes because of performance degradation, limited range, non-central administration, and weak encryption.

**Step 2 – Concept Map**

Create a “concept map” as a final report of your organized research.

* Use the diagram in the introduction as a starting point.
* You should have six (6) first level topics from “Application Software”   
  to “Network Connectivity”
* Each first level topic should have at least three (3) sub-topics
* Each sub-topic should be supported by a number of facts / items of interest

Select the best and most interesting information from your organized research.

* Summarize and edit your information to fit on the concept map.

Upload your Research Notes and Concept Map to your GitHub Repository

* Your concept map can be created using: Smart Ideas, Prezi, PowerPoint or other   
  similar applications.
* Option1: Create and upload a PDF of your concept map
* Option2: Include a link to your Concept Map in your Student Questions
  + Make sure that your link is Sharable so Mr. Nestor can open your map

**Appendix A**

|  |  |  |
| --- | --- | --- |
| **Operating System** | **Student 1** | **Student 2** |
| Ubuntu  (Linux) |  |  |
| z/OS  (IBM) |  |  |
| Solaris  (Oracle) |  |  |
| HP-UX  (Hewlett Packard) |  |  |
| Windows NT  (Windows Server) |  |  |
| Red Hat Enterprise (IBM Summit) |  |  |
| QNX  (Blackberry) |  |  |
| VxWorks  (Wind River) |  |  |
| AOSP  (Android Alphabet) |  |  |
| Ubuntu  (Linux) |  |  |
| z/OS  (IBM) |  |  |
| Solaris  (Oracle) |  |  |
| HP-UX  (Hewlett Packard) |  |  |
| Windows NT  (Windows Server) |  |  |
| Red Hat Enterprise (IBM Summit) |  |  |
| QNX  (Blackberry) |  |  |
| VxWorks  (Wind River) |  |  |
| AOSP  (Android Alphabet) |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |