
University Of Portsmouth
BSc (Hons) Computer Science
First Year

Core Computing Concepts

M30220

September 2022 - May 2023

20 Credits

Thomas Boxall
up2108121@myport.ac.uk

Contents

I	Video Essay	2
1	Introduction To Module	3
2	Reviewing Previous Work	4
II	Web	5
3	Introduction & Markup	6
4	Style	8
5	URLs and Images	9
III	Security	13
6	LECTURE: Week 0 & Introduction	14
7	WEEK 1: Security 101	15
8	WEEK 2: Access Control	20
9	WEEK 3: Cryptography	24
10	WEEK 4: Risk Management	27
11	WEEK 5: Commom Software Errors	30
IV	Key Concepts of Usability (KCU)	34
12	Introduction to HCI & Usability	35
13	Usability Heuristics	39
14	Usability Testing	42
15	Designing for Accessibility	46
16	Standards	49

Item I

Video Essay

Page 1

Introduction To Module

📅 27-09-22

🕒 13:00

🎓 Nadim

📍 RB LT1

This module is split into four items. The first of these is a video coursework project. The second, third and fourth are combined into an end of year exam. Each item is worth 25% of the overall module grade, therefore the end of year exam is worth 75%.

This module was created because the University doesn't do modules which are smaller than 20 credits and none of the items are big enough to be their own module.

To pass the module, you need to score at least 40% overall, not in each individual item.

The four items are taught by different lecturers and are shown below

1. Video coursework
2. Web
3. Security
4. Either UXD or DB (which we do is decided for us)

Each item will be introduced to us when we start that item.

Item 1: Coursework

This is due at 11pm on 16th December 2022. It is to be uploaded to YouTube, with a link to the video put in a PDF document which is uploaded to Moodle.

This item is able to be done either in groups or by individuals, it should be very easy to get a good grade in it.

The task is to select a conversation and analyse the conversation, using supporting research and references.

The video should be at most 4 minutes long, it can be a mixture of different takes edited together. There are a number of different pieces of editing software available on App-Somewhere.

More Information

More information for this coursework can be found on a Google Doc which is linked from the Moodle page. This Google Doc links to the official University Coursework information document, the conversations and outlines the lecture plan for the first half of TB1.

There is a one lecture per week and an optional drop in session per week. The drop in sessions are primarily there to answer quick questions.

Page 2

Reviewing Previous Work

📅 04-10-22

🕒 13:00

🎓 Nadim

📍 RB LT1

Historically, one of the biggest weaknesses to previous coursework submissions was the lack of knowledge of how it was graded.

For an animated video, look at Powtoons.

You are able to use AI voice generators to speak the script, however this is a risk as its not your voice on your submission. If you do use an AI voice, you must submit the script as a PDF to prove it is your own work.

The following list are things which were included in examples that I think are the attributes of high scoring videos

- Present the video as an argument, with one side then respond to it from the other perspective;
- Lift quotes out of the conversations and question, elaborate and research around them;
- Use evidence for all points

Th argument analysis included after each conversation is new for this year and the use of it won't loose or gain marks. It is there to give guidance for those who are unsure of where to start otherwise.

Item II

Web

Page 3

Introduction & Markup

📅 08-11-22

🕒 13:00

🎓 Rich and Co.

📍 RB LT1

Introduction to Item II

There are a number of different lecturers on this module: Rich, Matt and Kirsten. The exam will be computer based however not all computer marked. It will comprise mostly of multiple choice questions which will test knowledge of modern HTML and CSS. The multiple choice answers will be evil. The best practice is preparation. Exam date and time will be in January and will be announced on timetable at some point.

There is a Google Doc linked from Moodle which contains all the information and resources about this item. This document contains, pre session, during session and post session work.

There are drop ins on Thursdays in the FTC, these are compulsory.

There is a channel on the Discord Server (#ccc-web) where support can be sought.

The recommended book is available electronically through the library. One of the authors, Remy, has delivered guest lectures at the University.

Online Resources

Look at Mozilla Developer Network, add MDN to the end of any Google query about web development and their resources will come up.

Do not use W3Schools. It is bad.

Markup

Markup comes from the days of editors hand writing articles to be printed then annotating that with styles. This document then goes to a Typesetter who would design the content based off of the editors markings, hence markup.

HyperText Markup Language (HTML) is a form of markup, which is non-linear. It is a series of opening and closing tags, which together make an element. Elements can have attributes which provide more information on them or the way in which they should behave.

HTML Introduction

HTML5, the latest and most up to date version, should always start with the line `<!doctype html>`. This will tell the browser that the page is to be rendered as a HTML5 document.

A HTML document is comprised of two sections, a `<head>` and a `<body>`. In HTML5, the two sections do not need to be marked out as different sections, once you have specified that the document is HTML5 then the renderer is able to infer the difference.

<head>

This contains information about the document. Elements which you might see include `<title>` which defines the title of the page and `<meta>` which provides additional information about the webpage. Nothing in the head element is rendered.

<body>

This contains the content of the page. Numerous different tags are available within this to define the style of the content.

Markup

There are two types of Markup.

Procedural

This defines what to do and how it looks. It does not define why to do it.

Descriptive

This says what it means, not how it looks or what to do.

This is stratified (separates content from presentation), dynamic (different presentation to suit circumstances) and semantic (enables machine processing).

This means that we use descriptive markup, with semantic value, improving information quality and consequently styling of our pages must be achieved outside HTML.

Page 4

Style

📅 15-11-22

🕒 13:00

🎓 Rich & Co

📍 RB LT1

Cascading Style Sheets

Cascading Style Sheets (CSS) have been around since about 1997. They are a W3C standard for styling HTML and take the form of text files. The files contain rules which users define.

CSS is comprised of a number of rules.

```
LANGUAGE: CSS
1 p{
2   background: red;
3   color: white;
4   padding: 1em;
5 }
```

The rule above will turn the background colour to red, the text colour to white and give a padding on all sides of 1em to every p element in the page.

Selectors

There are a number of different ways in which we can define what elements in a HTML document we want to target with a given CSS rule.

- `p{}` will target all p elements within the document. This is the same for any other element when the rule is written this way.
- `*{}` will target *everything* in that HTML document.
- `#myid{}` will target the elements with the id of `myid`. This is the same for any other ids used in the same way.
- `.myclass{}` will target all the elements with the class of `myclass`. This is the same for any other classes used in the same way.
- `classOne, classTwo` will target both `classOne` and `classTwo`. This is useful for when multiple components on a HTML page need styling in the same way.

Page 5

URLs and Images

📅 22-11-22

🕒 13:00

🎓 Matt & Co

📍 RB LT1

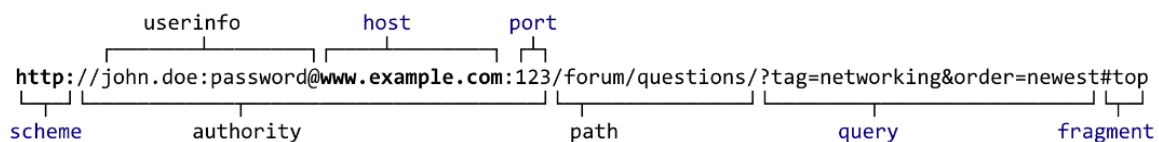
URLs

Uniform Resource Locators (URLs), a subset of Uniform Resource Indicators (URIs), allow us to navigate throughout the internet. They take the following form:

```
https://port.ac.uk/
```

```
http://www.example.com/forum/questions/?tag=networking&order=newest#top
```

They can be typed into an address bar, hyperlinked (using the `<a>` `` tags) or used as the `src` attribute on elements.



URL Structure

The `<a>` tag

The text between the two tags is what is displayed to the user. The `href` attribute specifies the resource. For example

LANGUAGE: HTML

```
1 <a href="https://www.example.com/">An Example Website</a>
```

To ensure that the webpages created are accessible to assistive technologies (ie screen readers), do not hyperlink the phrase “click here”, rather hyperlink a more descriptive phrase for example “on our website”.

target Attribute

The `target` attribute can be used to specify *where* to open the link. This can annoy users.

```
<a href="https://bbc.co.uk/news" target="_blank">BBC News</a> opens in new tab
```

URL Types

Absolute URLs are complete paths to resources, these can be on the same web server or on different web servers.

```
/on/the/same/web/server.html
```

```
https://on/a/different/web/server.html
```

Relative URLs refer to resources on the same server. The resource requested is ‘navigated’ to from the current working directory.

```
sections/contact.html
../index.html
```

Fragments

A fragment can be used to link to a specific part of a page. This can be used to scroll to particular parts of the page.

LANGUAGE: HTML

```
1 <a href="blurb.html#ch2">Chapter 2</a>
```

The example above will scroll the browser to the element with the id of `ch2`.

Images

Images are defined using the `` tag. Note that this tag doesn't require a closing tag. A basic, *valid* `` tag requires the `src` and `alt` attributes, to define the source file it will render and alt-text which can be used by screen readers.

LANGUAGE: HTML

```
1 
```

By default, images are displayed at full size. Images can be resized using CSS. For example, to set all images to have width of `30rem` you would use:

LANGUAGE: CSS

```
1 img {
2   width: 30rem;
3 }
```

The other types of CSS selectors (class, id, etc) can be used for image styling as well.

We can use the `float` CSS property to float images to the left or the right of other elements. It is no longer considered best practice to use this method of alignment as the `flex` layout model achieves the same result while being more predictable and controllable.

Captions

Captions can be added to images as seen below.

LANGUAGE: HTML

```
1 <figure>
2   
6
7   <figcaption>This is a teapot.</figcaption>
8 </figure>
```

Dynamically changing image widths

In addition to the `src` attribute, we can use the `srcset` attribute to provide a list of alternate images which can be dynamically chosen based on image & display size. We can add width hints (these come after the file name) which helps the browser select the most appropriate sized image.

LANGUAGE: HTML

```
1 
```

We can achieve a similar effect using CSS.

5.0.1 An alternative to <a>

The `<picture>` element allows us to use different images in different situations. It contains multiple `<source>` elements each defining a rule for when it is relevant and then specifies which image to use via its `srcset` attribute.

LANGUAGE: HTML

```
1 <picture>
2   <!-- On the biggest screens a large square image -->
3   <source
4     media="(min-width: 1600px)"
5     srcset="
6       i/elmer/800x800.png 1x,
7       i/elmer/1600x1600.png 2x,
8       i/elmer/3200x3200.png 3x">
9   <!-- On smaller screens a sidebar-style image -->
10  <source
11    media="(min-width: 800px)"
12    srcset="
13      i/elmer/200x800.png 1x,
14      i/elmer/400x1600.png 2x,
15      i/elmer/800x3200.png 3x">
16  <!-- Default is a wide masthead shape -->
17  <source
18    srcset="
19      i/elmer/800x200.png 1x,
20      i/elmer/1600x400.png 2x,
21      i/elmer/3200x800.png 3x">
22  <!-- fallback image (required)-->
23  
26 </picture>
```

Using the `<picture>` tag also allows us to specify a number of alternative images so that browsers with different capabilities can show their preferred image format

LANGUAGE: HTML

```
1 <picture>
2   <source
3     type="image/webp"
4     srcset="i/uop/logo.webp">
5   <source
6     type="image/svg+xml"
```

```
7   srcset="i/uop/logo.svg">
8 <source
9   type="image/png"
10  srcset="i/uop/logo.png">
11 
14 </picture>
```

Videos

We can include videos in webpages in a similar ways to images. It is best practice to specify multiple formats as for legal reasons not all browsers can support all video formats


```
LANGUAGE: HTML
1 <video
2   controls=true
3   muted
4   style="width: 100%"
5   id="v1">
6   <source src="i/vid/museum.ogv" type='video/ogv'></source>
7   <source src="i/vid/museum.mp4" type='video/mp4'></source>
8   <source src="i/vid/museum.mov" type='video/quicktime'></source>
9   <p>Sorry, your browser doesn't do video.</p>
10 </video>
```


Item III

Security

Page 6

LECTURE: Week 0 & Introduction

 24-01-23

 13:00

 David

 RB LT1

This sub-module will use Flipped Learning. There are a collection of videos on Moodle which we will need to watch each week. Lectures will be used to review previous weeks content and metrics based off of quiz scores. The quizzes do not feed into our final grade and can be found on Moodle. Also on Moodle, is a 'hotspots' activity which provides additional information which is needed to complete the quiz.

The lectures will be adapted each week to allow review and consolidation of content.

Page 7

WEEK 1: Security 101

📅 25-01-23

👁 Flipped Learning Lecture

Information Security

The preservation of confidentiality, integrity and availability of information

Confidentiality

The property that information is not disclosed to unauthorised individuals, entities or processes.

Integrity

The property of safeguarding the accuracy and completeness of assets.

Availability

The property of being accessible and usable upon demand by an authorised entity.

Often when introducing systems which protect one of these characteristics, for example confidentiality, there is a trade off that another characteristic will lessen, for example integrity.

Assets

Anything that has value to the organisation, its business operations and its continuity.

Threat

A potential cause of an incident that may result in harm to a system or organisation.

Threats can be both internal (employee leaking confidential data) and external (criminal accessing data for financial gain).

Vulnerability

A weakness of an asset; group of assets; or information system that can be exploited by one or more threats.

Impact

The result of an information security incident, caused by a threat, which affects assets.

Impacts can include the monetary loss incurred directly due to the loss of information; cost of responding to the incident; fines incurred for failing to adequately and reasonably failing to prevent such an incident; or to the brand or reputation of the brand which ultimately leads to loss of customers.

Risk

The potential that a given threat will exploit vulnerabilities of an asset or group of assets and thereby cause harm to the organisation.

Investment in security isn't like other investments within an organisation as the investments are made to prevent loss of income as a result of an incident where other investments may be made to increase income. The decision to invest in security is often made when weighing up: the cost of dealing with a small security incident frequently versus the cost of dealing with one big security incident which will probably never happen but if it did then it would be catastrophic for the organisation. The process organisations go through to identify, assess and control risks is called *risk management*.

Assets

Information assets are typically among the highest value assets of an organisation. ISO 27005 defines primary and supporting assets. Where supporting assets are only of interest when assessing information security risks where compromises to the supporting assets may adversely impact the primary assets.

Primary assets	Secondary Assets
Business processes & Activities	Hardware
Information	Software
	Network
	Personnel
	Site
	Organisation's Structure

Information

Information assets are typically of the highest value to the organisation, this is especially the case for *business critical information* without which the business would cease to operate. This is also the case of *personal information*, which is the data of employees and customers which by law must be adequately protected. *Strategic Information* should be protected, as releasing this information may lose that business a competitive advantage. *High-cost information* is information whose gathering, storage, processing and transmission require a long time and/or involve a high acquisition cost, the impact of losing this information may be that the high costs will be duplicated.

Business Processes

These are processes that contain secret processes; processes involving proprietary technology; processes that if modified can greatly affect the accomplishment of the organi-

sation's mission; or processes that are necessary for the organisation to comply with contractual, legal or regulatory requirements.

Significant adverse impacts can arise from the failure to adequately document or protect those business processes. Organisations will commonly find out when an employee is off sick, on holiday or leaves the company that the processes they oversaw are not sufficiently documented.

Hardware

This is the physical technology that: houses and executes the software; stores and carries the data; or provides the interface for data entry/ removal from the system. This includes, but is not limited to: desktops, servers, laptops, scanners, keyboards, monitors and hard drives.

Physical security is just as important as physical access can often mean information can be readily extracted. Whilst the hardware may be relatively cheap, the information stored, processed or captured on it may be worth millions.

Software

Software, within the information security sphere, comprises of applications, operating systems, assorted command utilities. Software is arguably the most difficult information security component to secure as software development is often under resourced therefore information security is often only added as an afterthought rather than being embedded as an integral part. When designing the specification for new software, the security requirements should be included up-front, at the same time as core functionality. The exploitation of software errors in software programming accounts for a substantial proportion of attacks on information.

Networks

Distributed hardware and software components are connected through networks by routers, switches, relays, firewalls, etc. Collectively these manage the effective transmission of information between interconnected computing devices. Connections from within the network out onto the internet and to other partner networks, expose systems to attacks. Policies, as well as architectural and technical responses can be put in place to reduce the likely hood of these attacks succeeding. This can be done by examining ports, protocols and packets at the network perimeter to ensure that only the data which is required to support the business to function is being exchanged. Firewalls should be used to create a 'buffer zone' between the internal business network and outside untrusted networks (including the internet). Firewalls should be configured such that everything is denied by default and a white list is implemented which only allows the traffic through which is required. Inbound and outbound data at the perimeter should also be scanned for malicious content.

As well as protecting the perimeter, its also important to protect the internal network. Part of this is ensuring there is no direct routing between internal and external services. Network traffic should be monitored to detect (and then be able to react to) attempted or actual network attacks. Critical business systems should be segregated within the network and appropriate controls should be put in place to access these. Appropriate access controls to both wireless access points and to other hardware should be secure and *should not* be left as default passwords. Network intrusion, prevention and detection tools should be deployed on the network and configured by qualified staff. Systems should automatically generate alerts which staff can manage as part of an incident response plan.

Personnel

Personnel is an often over-looked component of securing an information system. They themselves are susceptible to numerous vulnerabilities. People make mistakes on a daily basis that compromise information assets. They are also susceptible to social engineering, bribery and blackmail. Due to this, it's important that all staff are adequately trained in how to perform their duties securely. Users have a critical role to play in their organisations security.

Systematic delivery of training should be deployed to ensure employees are trained and to help to enforce a security conscious culture. Organisations should develop a comprehensive set of policies covering security and computer use topics, these policies should be written using plain business terms and reduce the use of technical jargon. New employees should be made aware of policies and the companies procedures as part of their induction. The effectiveness and awareness of training should be monitored. The organisation should strive to promote a security conscious culture where staff feel empowered to voice their concerns. Organisations should also be aware that mistakes will be made by even the most security conscious of individuals.

Subject or Object

Object of the attack is the entity which is being attacked, the target.

Subject of the attack is the entity carrying out the attack against the target.

Subject attacks the object.

Computers can be compromised which can lead to it carrying out an attack on another machine. A person might be blackmailed or bribed to carry out an attack. In both of these examples, the entity (computer/ person) is both the subject and object of an attack.

Information Security Governance

Information Security Governance

How organisations control, direct & communicate their cyber risk management activities.

This will include a collection of policies, including but not limited to: overarching information security policy; ICT acceptable use policy; and other issue specific policies eg remote working. These policies must be continually reviewed and revised to keep up to date with the business needs and continually changing threats/ vulnerabilities.

To remain viable, the security policies must state: the individual responsible for the policy; schedule of review; method of making recommendations for reviews; specific policy issuance and revision dates.

Policy

A principle or rule to guide decisions and achieve rational outcomes. Should be broadly applicable to the widest possible set of circumstances and contexts supporting employees in deciding the most appropriate course of action in any given situation.

Procedures

A list of steps that constitute instructions for performing some action or accomplishing some task. These cannot exhaust all possible actions undertaken by an employee.

Standards

Detailed statements, quantifying what must be done to comply with policy. For example, 'passwords must contain a mixture of 8 numbers, letters and special characters and they should be changed if compromised'. Compliance with standards is also mandatory, they should state what should be done and how it should be achieved.

Guidelines

A set of recommended actions to assist in complying with policy.

Disseminating Policies

Policies should be promoted/ supported by a security education, training and awareness (SETA) programme that helps employees do their jobs securely.

Not everyone in the organisation needs a formal degree or certification in information security, however some roles may require certain employees to hold information security academic qualifications or industry certification.

Everyone in an organisation needs to be trained in information security. Training provides employees with hands on instruction with regards to their specific jobs which enables them to perform their duties securely. Management of information security can be developed in-house or outsourced to outside training providers. Training will often make use of safer environments rather than the production systems.

Security awareness is not intended to teach something new. Instead it aims to keep elements of information security at the forefront of employees minds, this is information which they already possess due to their education and training. Materials may be disseminated in a variety of creative means, such as posters, mouse mats or even coffee mugs.

NCSC Guidance

Good security governance should clearly link security activities to your organisation's goals and priorities; identify the individuals at all levels who are responsible for making security decisions & empower them to do so; ensure accountability for decisions; ensure that feedback is provided to decision-makers on the impact of their choices; and fit into an organisation's wider approach to governance. Security needs to be considered alongside other business priorities such as health and safety or financial governance.

Incidents Happen

Incidents will happen, we may be able to considerably reduce the likelihood of an incident, however not remove it completely, we can further reduce the risk by minimising the impact of the incident. This is done through incident response management.

Page 8

WEEK 2: Access Control

📅 2023-02-01

👁 Flipped Learning Lecture

Identification & Authentication

Users must be instructed to enable user-specific access controls and given individual accountability for their actions.

Claimed identities must be authenticated. This is the first line of defence for the system and safeguards against unauthorised use (internal and external). Traditional passwords are the most common means of authentication in digital systems, they are conceptually simple for designers & users, and can provide good protection if used correctly; however the protection they provide is often compromised by users.

Passwords

Passwords have a number of vulnerabilities: it is easy to badly select one; they get written down; infrequently or never changed; same password used for multiple systems; only require them at the start of a session; they can be forgotten; and they can be shared. The traditional defence against password guessing (lock the user out after three failed password attempts) enables a form of Denial of Service (whereby the attacker disrupts availability by deliberately locking out users).

NCSC Password Guidance

The National Centre for Cyber Security encourage organisations' to reduce reliance on users recall of large numbers of complex passwords. They have published 6 tips.

1. Reduce your organisations reliance on passwords
2. Implement technical solutions
3. Protect all passwords
4. Help users cope with password overload
5. Help users to generate better passwords
6. Use training to support key messages

Access Control

Identity

The properties of an individual or resource that can be used to identify uniquely one individual or resource that can be used to identify uniquely one individual or resource.

Authentication

Ensuring that the identity of a subject or resource is the one claimed.

Authorisation

The process of checking the authentication of an individual or resource to establish their authorised use of, or access to information or other assets.

Accounting

Ensures that user activities can be tracked back to them.

Audit

Formal or informal review of actions, processes, policies and procedures.

Compliance

Working in accordance with the actions, processes, policies and procedures laid down necessarily having independent reviews.

Authentication

Factors of Authentication

There are three widely used authentication mechanisms (factors).

Something a supplicant knows which includes: personal identification numbers (PIN); passwords; passphrases; and security questions/ answers.

Something a supplicant has which includes: dumb cards (magnetic stripe ID cards and ATM cards); smart cards (chip and pin cards); and security token (key fob, card reader etc.)

Something a supplicant is which includes: fingerprint; palmprint; retina/iris scanner; voice; keyboard kinetic measurements.

Strong Authentication

Strong customer authentication is a procedure based on the use of two or more of the following elements - categorised as knowledge, ownership and inherence: something only the supplicant knows; something only the supplicant possesses; and something the supplicant is. In addition, the elements must be mutually independent (which means breach of one doesn't compromise the other).

Biometrics

Biometric technologies are evaluated on three basic criteria.

False Rejection Rate (FRR) is the percentage of identification instances in which authorised users are denied access. This is a Type I error.

False Accept Rate (FAR) is the percentage of identification instances in which unauthorised users are allowed access. This is a Type II error.

Crossover Error Rate (CER) is the level at which the number of false rejections equals the false acceptances. This is the Equal Error Rate (EER).

Biometric System Requirements

There are a number of requirements which biometric systems have.

Universality - every individual in the population should possess the trait.

Distinctiveness - the ability of the trait to sufficiently differentiate between any two persons.

Persistence - the trait shouldn't change too much over time on the individual in question.

Collectability - the trait should be easy to collect or be measurable.

Performance - within a variety of operational and environmental conditions, high recognition accuracy and speed should be achievable.

Acceptability - the biometric identifier should have a wide public acceptance and the device used for measurement should be harmless.

Circumvention - it should be difficult to spoof the characteristic using fraudulent methods.

Access Control

Access privileges are specified and subjects' access to objects are determined through a security policy. There are a number of different access control policies.

Discretionary Access Control Policy (DAC) - controls access based on identity.

Mandatory Access Control Policy (MAC) - controls access based upon security labels.

Role-Based Access Control Policy (RBAC) - controls access based on roles.

Attribute-Based Access Control Policy (ABAC) - controls access based on attributes.

File System Permissions

By default, file systems come with four permissions: read, write, execute, and none of the above.

Privileged Access

The conventional name for the user with top-level access is system or application administrator (sysadmin for short). They can: enrol new users; modify user access right; remove user access rights. They can also modify groups or levels of privilege, rebuild the system, erase data, grant or deny access to applications; change passwords; and even alter or destroy event logging or auditing data.

These accounts have great power and wide-ranging capabilities and their use must be tightly controlled and safeguarded. Their power to disrupt operations, accidental or otherwise is enormous.

Security Policies

Access privileges are specified and subjects' access to objects are determined through a security policy. There are a number of different access control policies.

Discretionary Access Control (DAC)

This controls access based on identity. It is at the discretion of the owner and the permissions are often shown as a matrix.

Alternatively an Access Control List (ACL) can be used. These store the access rights to an object within the list.

Alternatively to either of the above, a Capability List (CL or C-List) can be used to store access rights. These store all the access to the rights to an object within a subjects unique list.

Mandatory Access Control (MAC)

This controls access based upon security labels. There is less individual in this as the OS has overall control. Rules are used for defining how the subject can behave. It uses sensitivity (or security) labels to define access rights (these may include: top secret; secret; classified; and unclassified).

MAC makes use of a number of access control models.

Dell-LaPadula Confidentiality Model which provides a "no read up no write down" system. This enables users to read everything below them, write to their level of security and have no access to anything above them.

Biba Integrity model which provides a "no write up, no read down" system.

Clark-Wilson Integrity Model which has a number of rules: no changes by unauthorised subjects; and no unauthorised changes by authorised subjects. Maintenance of internal consistency (system does what is expected without exception) and external consistency (data in the system is consistent with similar data in outside world) is important here.

Graham-Denning Access Control Model has three key parts: objects, subjects and rights.

Brewer-Nash Model in which subjects can only access one of two conflicting sets of data (which prevents conflicts of interest).

Role Based Access Control (RBAC)

This policy is newer than DAC and MAC. It is a centrally administered set of controls through which permissions are assigned based on roles. Users who perform a similar function are grouped together (for example Moodle has student access and lecturer access). It is a useful model for companies with high employee turnover.

Each user can be a member of many roles and each role can have many users as members. A permission can be assigned to many roles. Each role can have many permissions.

Attribute Based Access Control (ABAC)

This generalises access control based on the role attribute of users. It uses various other attributes of the users including their environment and information assets to determine permissions.

Page 9

WEEK 3: Cryptography

📅 2023-02-08

👁 Flipped Learning Slides

Cryptography is a way of turning *plaintext* (the text to be encrypted) into *ciphertext* (an unreadable version that can later be turned back into plaintext).

Encrypting something links four elements together: the plaintext m ; the ciphertext c ; the key k ; and the algorithm E .

$$c = E_k(m)$$

Modern cryptographic algorithms abide by the following principles

- Large enough key space to resist exhaustive search
- Resistant to frequency analysis
- Small change in plaintext results in large change in ciphertext
- Security depends only on secrecy of key and not on secrecy of algorithm (Kerckhoff's principle)

Cryptographic Algorithms

Symmetric

Encryption and Decryption both use the same "secret key". This means the encryption methods can be extremely efficient, requiring minimal processing. Both the sender and receiver must possess the encryption key; and if either copy of the key is compromised, and intermediate can decrypt and read messages.

Asymmetric

Asymmetric encryption uses two different, but related, keys to encrypt/ decrypt messages. This works as follows: if key A encrypts the message, only key B can decrypt. This methodology has the highest value when one key serves as the private key and the other serves as the public key. It is typically used to encrypt a symmetric session key rather than the plaintext message(s).

Caesar Cipher

The Caesar Cipher is one of the simplest forms of a cryptographic algorithm. In it, a 'wheel of letters' are turned and letters simply substituted, for example A becomes B, B becomes C and so on.

This cipher can be very easily brute forced as there is a maximum of 25 possible options to try, and using common letter (E T A) frequency, a good guess can be made relatively quickly.

Symmetric Cryptosystem

A symmetric cryptosystem starts with the plaintext being encrypted with the shared key. This produces the ciphertext which is transmitted to the recipient. The recipient is then able to decrypt the ciphertext using the shared key which produces plaintext.

Data Encryption Standard (DES), Triple DES (3DES) and Advanced Encryption Standard (AES) are all examples of Symmetric Encryption.

With symmetric encryption, the primary challenge is distribution of keys. In general we will need $n \times (n - 1)/2$ keys where n is the number of parties who want to communicate.

Diffie-Hellman Key Exchange

The Diffie-Hellman (DH) key exchange uses asymmetric encryption to exchange session keys, these are limited use symmetric keys for temporary communication. This allows two entities to conduct, quick efficient, secure communication based on symmetric encryption which is more efficient than asymmetric encryption for sending message.

DH based key establishment is incorporated into a number of standard protocols: TLS/SSL and IPSec. DH based key establishment is also used in a variety of applications: GlobalProtect VPN; and WhatsApp.

Asymmetric Cryptosystem

An asymmetric cryptosystem works by the symmetric encryption key being used to encrypt the plaintext. This is then transmitted to the recipient where it is held until the encryption key is received. The encryption key (which has already been used to encrypt the plaintext) is passed through an encryption algorithm that uses the recipient's public key. This, now encrypted, message is transmitted to the recipient where they can use their private key to decrypt it. The recipient now has the symmetric encryption key and can use that to decrypt the ciphertext they received into the plaintext.

RSA and Elliptic Curves are examples of Asymmetric Encryption.

RSA

RSA is the most famous asymmetric cryptographic algorithm, developed by Rivest, Shamir and Adleman in 1978. It is based on number-theoretical properties of natural numbers.

The steps below show RSA in a nutshell.

1. Choose two large primes p and q , and calculate $n = p \times q$
2. From n follow some maths steps to calculate the value e and d
3. Publish n and e , keep d a secret and destroy p and q
4. Encryption of m is now $c = m^e \pmod n$
5. Decryption of c is then $m = c^d \pmod n$

Example encryption and decryption

Shown below is an example encryption of the plaintext GAGA

1. Encode the letters as ASCII numbers 71 65 71 65
2. Encrypt each letter in turn using $c = m^e \pmod n$ assuming $e = 131$ and $n = 232$. ($d = 11$ but this is only known to the receiver).

3. This gives $71^{131} \pmod{323} = 10$ and $65^{131} \pmod{323} = 126$.

4. The message that gets transmitted is 10 126 10 126.

Shown below is the decryption of the message encrypted above.

1. The receiver knows $d = 11$ and $n = 323$ and the algorithm $m = c^d \pmod{n}$

2. $10^{11} \pmod{323} = 71$ and $126^{11} \pmod{323} = 65$

3. Convert 71 and 65 to letters using ASCII

4. Results in G and A

The examples shown here use very small primes to make the calculations easy. In reality, the prime numbers used are much bigger.

Page 10

WEEK 4: Risk Management

📅 2023-02-19

👁 Flipped Learning Slides

Component driven approaches to security require the risk analysis to assess three elements of risk: threat (the individual, group or circumstance which causes a given impact to occur), vulnerability and impact. The purpose of assessing threat is to improve the assessment of how likely a risk is to be realised.

We manage risks in our everyday lives, for example deciding if to wait for the *green man* or stepping out when it is clear.

The higher likelihood and impact of a risk, the higher risk the activity is.

Information Security Risk Methods and Frameworks

There are a number of Risk Assessment Methods/ Frameworks

- NIST 800-30
- ISO/ IEC 27005
- ISACA COBIT
- ISF IRAM 2
- HMG Information Assurance Standard 1 2
- Octave Allegro
- ISACA COBIT 5

Equally, there are a number of Information Security Management Frameworks

- NIST CSF
- ISO/ IEC 27000 series

Risk Assessment Steps

1. Identify risks

- Identification of Assets
- Identification of Threats
- Identification of Existing Controls
- Identification of Vulnerabilities
- Identification of Consequences

2. Analyse risks

- Assessment of consequences

- Assessment of incident likelihood
- Level of risk determination

3. Treat risks

- Risk modification
- Risk retention
- Risk avoidance
- Risk sharing

4. Monitor & review

Qualitative Risk Analysis

Qualitative risk analysis uses a scale of qualifying attributes to determine the magnitude of the consequences/ likelihood (the scale could be: very low, low, medium, high, very high). An advantage of this is that it is easy to understand by all relevant personnel. A disadvantage is that it is very subjective on the choice of scale. Qualitative risk analysis may be used: as an initial screening to identify risks requiring detailed analysis; where this analysis is sufficient for decisions; or where numerical data resources are inadequate for quantitative analysis.

Quantitative Risk Analysis

Quantitative risk analysis uses a scale of objective numerical values for consequences/ likelihood values. It uses data from a variety of sources and the quality of analysis depends on the accuracy/ completeness of numerical data. Typically, historical incident data is used which has the advantage of relating directly into information security objectives & the concerns of that organisation; however it has a number of disadvantages: there is a lack of data on new risks; and accurate or missing data in general can create an illusion or worth or accuracy of risk assessment. Uncertainty and variability of consequences/ likelihood are to be considered and communicated.

Cost Benefit Analysis

Cost Benefit Analysis (*CBA*) can be calculated using the following equation

$$CBA = ALE_{prior} - (ALE_{post} + ACS)$$

where *ACS* is the annualised cost of safeguard; and *ALE* is the annualised cost of safeguard. *ALE* (annualised loss expectancy) can be calculated using the following equation

$$ALE = SLE \times ARO$$

where *SRE* is the single loss expectancy; and *ARO* is the annualised rate of occurrence.

Risk Treatment Options

Retain/ Accept - organisation may tolerate (but not ignore) the risk.

Avoid/ Terminate - organisation may decide not to do the thing that incurs risk.

Share/ Transfer - transfer risk via an insurance policy or a third party.

Modify/ Reduce - adopt controls to lower the current levels of risk (by reducing likelihood & impact).

Critical Appraisal of Risk Methods and Frameworks

The critical appraisal of risk methods and frameworks was originally produced by the NCSC so practitioners and decision makers can better understand and work with approaches available. It does not mean that existing risk frameworks should not be used, or that they are fundamentally ineffective. Whichever framework is used, effective outcomes won't be realised without thought or context.

Limitations

- Limits of a 'reductionist' approach
- Lack of variety
- Limits of a 'fixed state' approach
- Lack of feedback and control
- Losing risk signals in the 'security noise'
- System operation
- Information opacity
- Noise from misguided analysis
- Noise from bias
- Assumed determinability
- Abstraction through labelling
- The limits of using matrices
- Limits in the way uncertainty is presented
- The effect risk relationships have on impact
- The adverse effect of intervention
- Impacts are not limited to the scope of assessment
- The effect on time on risk

Page 11

WEEK 5: Commom Software Errors

📅 2023-02-22

👁 Flipped lecture slides

Software, the arguably most difficult information security component to secure comprises of applications, operating systems and assorted command utilities. Unfortunately, software development is often under resourced which can lead to information security being added as an afterthought rather than an integral part. The exploitation of software errors in software programming accounts for a substantial proportion of attacks on information.

Web Application Security Risks

Attack vectors may exist through your web app, each representing a risk. Paths from threat agents through attack vectors, security weaknesses, security controls, technical impacts, to business impacts may be trivial to find or exploit but also may be very difficult. The impact of each may be inconsequential to catastrophic. Risks combine a likelihood associated with a threat agent, attack vector, and security weakness with the estimated technical/ business impact.

Security Requirement Specifications

The security requirements must be part of the overall statement of requirements document from which the design is generated. Adding them later will almost always add complexity and cost to any project. Security should not be thought of as only being the need to defend against improper access and misuse, it also means

- defensive programming to make sure that only valid and accurate data is processed by the system
- proper functional testing to ensure it behaves as expected and within the design criteria
- methods to back up secure data against loss or damage
- adequate assurance of availability
- compliance with any legal and regulatory requirements
- effective auditing of activity.

Common Software Errors

The *Common Weakness Enumeration* (CWE) is a list of software security vulnerabilities and is a community-driven project maintained by MITRE (a non-profit research and development group). CWE provides a description for each vulnerability and a mitigation. MITRE has partnered with SANS institute to develop CWE/25, which lists the 25 most critical software vulnerabilities.

A similar list is provided in the *Open Web Application Security Project* (OWASP) top 10 project.

The CWE/25 and OWASP top 10 share many of the same vulnerabilities.

Cross Site Scripting

Cross Site Scripting

Inproper neutralisation of input during web page generation.

Cross Site Scripting (XSS) vulnerabilities occur when:

- Untrusted data enters a web app
- Web app dynamically generates web page containing untrusted data
- On page generation, the web app does not prevent entry of executable data (for example JavaScript, HTML tags, mouse events)
- Victim visits generated web page via browser, containing the script injected using the untrusted data
- Since script is from page sent by the web server, the victim's browser executes in the context of the web server's domain
- This violates the same origin policy, scripts in one domain are not to access resources or run code in a different domain.

Types of Cross Site Scripting

Type 1: Reflected XSS (Non-Persistent)

The server reflects data in HTTP request back to the client in a HTTP response. The attacker makes the victim supply bad content to the vulnerable web app, which is reflected back and executed by a victim's browser.

Type 2 Stored XSS (Persistent)

The app stores bad data in a database, message forum, visitor log, or other trusted data store. The dangerous data is later read back into the application and included in dynamic content.

Preventing Cross Site Scripting

Preventing XSS requires a separation of untrusted data from active browser content.

Rule 0 Never insert untrusted data except in allowed locations

Rule 1 HTML escape before inserting untrusted data into HTML element content

Rule 2 Attribute escape before inserting untrusted data into HTML common attributes

Rule 3 JavaScript escape before inserting untrusted data into JavaScript data values

Rule 3.1 HTML escape JSON values in an HTML context and read the data with `JSON.parse`

Rule 4 CSS escape and strictly validate before inserting untrusted data into HTML style property values

Rule 5 URL escape before inserting untrusted data into HTML

Rule 6 Sanitize HTML markup with a library designed for the job

Rule 7 Avoid JavaScript URLs

Rule 8 Prevent DOM-based XSS

Bonus 1 Use HTTP Only cookie flag

Bonus 2 Implement content security policy

Bonus 3 Use an auto-escaping template system

Bonus 4 Use the X-XSS-Protection response header

Bonus 5 Properly use modern JS frameworks

SQL Injection

SQL injection is where the software constructs all or part of an SQL command using externally-influenced input from an upstream component, but it does not neutralise or incorrectly neutralises special elements that could modify the intended SQL command when it is sent to a downstream component.

Secure Development & Deployment

Secure development is everyone's concern, it is important that all developers keep their security knowledge sharp and produce clean and maintainable code. It is also important to protect the code repository, secure the build and development pipeline and continually test your security. Plans should also be made for security flaws and development environments should be secure.

Securing the Development Environment

The ISO 27000 series contains requirements that live and development systems are kept separate. User training may also be safer to run in a non-live system and function & acceptance testing is required before releasing to production system.

It can be good practice to treat the development environment as if it is compromised all the time, which means we must separate business and development functions. It is also to have trust in the developers and verify their actions.

Acceptance Processes

Security testing needs to consider

- effectiveness of defensive coding
- protection against malware and code injection through interfaces
- backup and recovery of data
- access control
- auditing and behavioural analysis
- communications security
- resilience

Change Control

Any changes to software risks introducing new bugs or vulnerabilities. There should be a formal change control process that manages these risks. This should include: *separation of duties* (different people test and implement the system); *two person control* (a second person signs off on code changes, to reduce accidental or malicious flaws); and *using version control systems* (to be able to easily see the changes made and rollback changes quickly and easily).

Patching

Every piece of software contains bugs, the complexity and size of these applications means it is impossible to test 100% of the potential execution paths. Bugs have a variety of different impacts on confidentiality, integrity and availability. Once a bug is found and fixed, the software suppliers issue a patch that can be installed in order to remove the vulnerability. Patches should be rolled out at the earliest opportunity, however patches should be tested in a non-live environment before roll out. Vulnerabilities may already be known to attackers who may want to exploit them, hence the necessity for a quick patch rollout. Patches may also be reverse engineered to create new exploits by attackers.

Accreditation and Certification

Certification

Provision by an independent body of written assurance (a certificate) that the product, system or service in question meets specific requirements.

Accreditation

Formal recognition by an independent body, generally known as an accreditation body, that a certificate body operates according to international standard.

Accredited certification is typically mandated for safety/ security critical systems. A formal review process to approve information security architecture, policy and procedures before the new/ updated product/ service/ system is deployed/ used. This will typically require periodic review and re-accreditation.

Item IV

Key Concepts of Usability (KCU)

Page 12

Introduction to HCI & Usability

📅 2023-03-14

🕒 13:00

👤 John

📍 RB LT1

The format for this item will be a lecture on Tuesday where new content is delivered then a research task issued to be completed during the Thursday drop-in session. The research task is to be submitted on Moodle.

12.1 What is HCI?

Human Computer Interaction (HCI) has historically been known as the study of interaction between people and computers. As a result of the internet of things HCI is changing, it is shifting to become the study of interaction between people and machines.

The *Association for Computing Machinery* has defined HCI as "a discipline concerned with the design, evaluation and implementation of interactive computing systems for human use and with the study of major phenomena surrounding them".

12.1.1 Goals

HCI's major goal is to improve the interactions between people and computers, by extension machines; and to make computers more usable and receptive to the user's needs. HCI's long term goal is to design systems that minimise the barrier between the human's cognitive model of what they want to accomplish and the computer's understanding of the user's task.

12.1.2 Scope of HCI

HCI is very broad and encompasses many different fields of Computer Science, including: AI, Psychology, Sociology, Art, Design, etc.

12.1.3 Sub-Domains of HCI

There are three different subdomains of HCI, each has a different scope and focus however they are all concerned with the design and evaluation of the user interface.

Usability

Usability is concerned with making systems easy to learn, easy to use, limiting errors, and the severity of errors. Usability refers to how successfully a user can use a system to accomplish a specific goal and uses terms like: error rates, time to complete tasks, task failures, number of lookups made.

User Experience

User Experience encompasses the user's entire experience with an interface. This includes how well the user interface worked, how the user *expected* it to work, how the user felt about using it, and how the user feels about the system overall. User Experience uses terms like: satisfaction, intuitive, frustration, good experience, difficult, confusing.

User Centred Design

User-Centred Design (UCD) is an iterative design process in which designers focus on the users and their needs in each phase of the design process. Design teams involve users throughout the design process through research and design techniques; this enables them to create highly usable and accessible products for them.

12.1.4 Why Is HCI Important?

Badly designed interfaces can waste the user's time, cause frustration and lead to errors. Users will often leave websites or apps with bad interfaces in frustration.

For web apps and desktop apps, the principles of good user interface design have been well understood since the 1990's. However, applying these design principles is often neglected, which could be due to: prioritizing functionality over usability or user experience; budgetary and time constraints; or not having a clear part of the software development life cycle to consider usability and user experience.

12.1.5 HCI: A Solved Problem?

No. The rise in popularity of NUI methods (and the metaverse) is changing everything. Application of HCI to NUI and Metaverse is far from a solved problem. The popularity of NUI methods has now crossed a threshold for a paradigm shift, which could lead to a redesign of interfaces to suit NUI methods more (currently NUI methods often simulate mouse point and click). NUI methods are the foundations of the Metaverse.

12.2 Paradigms In HCI

There are four key paradigms in HCI, each different from one another.

12.2.1 Command Line Interfaces

Command Line Interfaces (CLIs) are text based. Users control the computer by typing in commands; this requires little processing power and can be extremely powerful, however they can take longer to learn than a GUI. Originally, most interfaces were CLIs and most modern devices will still have one. Internet of Things devices will often use CLIs. An example command execution in Linux is shown below.

```
user@bash: pwd
/home/myCSdata
user@bash: ls
file.txt
```

12.2.2 Graphical User Interfaces

Graphical User Interfaces (GUIs) are the desktop metaphor. They are based on point-and-click interaction which means they are not adapted for NUI input modalities, therefore NUI will often simulate point-and-click. Not much has changed with GUIs since 1984.

12.2.3 Natural User Interfaces

Natural User Interfaces (NUIs) are “a system for human-computer interaction that a user operates via intuitive actions related to natural, everyday human behaviour leveraging modalities like touch, gestures, or voice”.

A NUI mimics real-world interaction, however they are not fully developed so will often simulate point-and-click in the GUI. There are potential redesigns needed for the NUI method, which won't restrict to simulating point and click.

There are already a number of uses of NUI

Speech Recognition used in voice recognition in smart assistants. Users can ask questions, control home automation devices and media playback through their voice.

Brain-Machine Interface reads brain signals and translates them into actions within the computer systems. These have lots of possible applications in the health sector.

Touch Screen allows user to interact with device through touch of a finger. This is the most common form of NUI application.

Gesture Recognition tracking user motions and physical actions then using these as the input to computing devices. Eg, Nintendo Wii's controller.

Gaze Tracking uses user eye movements to estimate a position on screen for a mouse pointer.

12.2.4 Metaverse

Metaverse

Generally refers to the concept of a highly immersive virtual world where people can gather to socialise, play and work
- Merriam-Webster

A metaverse can either be immersion in VR or AR.

12.3 Interface Design

User Interface design is a subset of HCI. Designers must consider a variety of factors, including

- What people want and expect, physical limitations and abilities people possess;
- What people find enjoyable and attractive;
- How information processing systems work;
- Technical characteristics and limitations of the computer hardware and software must also be considered/

The *user interface* is the part of a computer and its software that people can see, hear, touch, talk to, or otherwise understand or direct. This includes the input and output of the machine: input is some form of communication of requirements (predominantly through point-and-click); output is the results of processing user's requirements (predominantly through the display screen).

Good user interface design will provide a mix of well-designed input and output mechanisms that satisfy the user's needs, capabilities and limitations in the most effective way possible. The best interface is one that is not noticed, rather it allows the user to focus on the information and task at hand, not the mechanisms used to present the information and perform the task.

The use of other human senses as part of the user interface is currently being developed.

12.4 Examples of Bad Interfaces

Windows 8 is an example of a bad interface. The designers tried to incorporate two operating systems into one. This led to it having double the complexity and requiring double the time to learn.

The Apple Watch interface has also been classified as a bad interface - this is down to its tiny targets which can be difficult to tap on, leading to miss-taps.

Page 13

Usability Heuristics

📅 2023-03-21

🕒 13:00

🎓 John

📍 RB LT1

Neilsen's heuristics are 10 principles for evaluating the usability of interfaces. They were created by Jakob Nielsen in 1990. The principles define important points in the composition of interfaces and should be considered when creating layouts.

The order the heuristics are listed in this lecture has been randomly chosen by John. This order may be different of that on the internet.

13.1 H1. Match Between Systems And The Real World

The system should speak the same language as the user. This means the language used should be familiar to the user (eg finance systems should use finance language), rather than the system using system-oriented language.

It should follow real-world conventions, making information appear in a natural and logical order.

An easy way to implement this would be to avoid using technical jargon and instead use language which the user can understand.

Another way which systems can be intuitively interpreted by users is to use metaphors, for example the use of the recycle bin icon for the rubbish folder.

Information should be presented to the user in a logical and natural order.

13.2 H2. Consistency Of Standards

Users should not have to wonder whether different words, situations, or actions mean different things in different context. All platforms should follow the same convention and should follow the principle of least surprise: similar things should look and act in a similar way & different things should look different.

Consistent language and graphics should be used, which should achieve the same visual appearance across the system, the same information/ controls should be in the same locations on all windows and colour themes should be the same.

Commands and actions should also have the same effect in equivalent situations.

13.3 H3. Visibility Of System Status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time.

Commonly this is achieved through a progress bar.

Response Time	Display
< 0.1 second	seems instantaneous
0.1-1 seconds	user notices, but no feedback needed
1-5 seconds	display busy cursor
> 5 seconds	display progress bar

13.4 H4. User Control And Freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Systems need to support undo and redo.

Users don't like to feel trapped by the computer - as many exits as possible should be provided, which should be clearly marked. For example: a cancel button; universal undo; interrupt; quit; restore to default.

Frequent violations of this heuristic can lead to a *pottery barn effect* (you break it, you bought it). We want to avoid users feeling tense while interacting with the system.

13.5 H5. Error Prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

People will make errors. People will also slip. Systems have to be designed to cope with errors, which could include double checking with a user before actioning an irreversible action. Systems should also be designed with errors in mind, this can be achieved through removing memory burden (through auto-fill); supporting undo and redo; or using constraints.

13.6 H6. Help Users Recognise, Diagnose, And Recover From Errors

Deal with errors in a positive manner, through being polite and speaking human readable language.

13.7 H7. Recognition Rather Than Recall

Minimize the user's memory load by making elements, actions and options visible. The user should not have to remember information from one part of the interface to another. Information required to use the design should be visible or easily retrievable when needed. The classic example of this is a CLI required users to remember exact commands whereas a GUI allows all the options to be presented to a user and then they can pick which they want.

13.8 H8. Flexibility And Efficiency Of Use

Accelerators (unseen by the novice user) may often speed up the interaction for the expert users, such as the system can cater to both inexperienced and experienced users. Allows for efficiency in expert users.

Accelerators can come in the form of: keyboard shortcuts, command abbreviations, bookmarks, history (of a CLI) or templates.

13.9 H9. Aesthetic And Minimalist Design

Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

13.10 H10. Help And Documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Users don't and won't read manuals, unless they are frustrated or in crisis. Help should be searchable, context-sensitive, task oriented, concrete, short.

Help is not a replacement for bad design.

Page 14

Usability Testing

📅 2023-03-28

🕒 13:00

🎓 John

📍 RB LT1

14.1 Usability Testing

Usability Testing

Evaluating usability of a web-page, app or other software by testing it with real users.

Often software development companies neglect usability testing or do very little testing. Quite often, usability testing is left to the end of the Software Development Lifecycle (SDLC). There are a number of reasons given for why usability testing is not completed:

- Not enough time
- Not enough money
- No expertise in doing itNo lab or location in which to perform it
- Don't know how to interpret results

Usability testing shouldn't be carried out by the software developers as quite often they will know the product too well, therefore they generally won't spot things that a new user would. Limiting testing is better than no testing and it is better than to do these tests earlier rather than later. Ideally, testing is iterative - it is done over and over many times.

14.1.1 When?

Usability tests should be done early in the SDLC and should be done often. They should often be repeated. The type of test completed may depend on how far along you are in the testing process. In the early stage this may just be a sketch on paper, the middle may be a comparison of UI designs and finally at the end of the design process this may be to verify the final usability of the UI. It is important to keep a historical record of usability results for each test.

14.1.2 Formative vs Summative Testing

Formative testing will feed into the design. This is typically done with some a prototype (Could just be a drawing on paper).

Summative will show a complete system and take measurements from it (eg how long it takes for users to complete a task).

Validation will show all the changes made so far and compare them to evaluate the end result of the overall usability of the system.

14.2 Testing Approaches

Traditionally, UI testing would have been expensive and used a scientific approach. An alternative testing approach, lost-our-lease, allows for cheap and often testing which is not scientific.

Item	Traditional Testing	Lost our Lease Testing
Number of users per test	8 or more	3 or 4
Recruiting Effort	Select carefully to match target audience	Anyone will do
Where to test	Specialist usability lab with observation room and one-way mirror	Any office or conference room
Who does testing	Experienced usability professional	Anyone
Advance planning	Tests have to be scheduled weeks in advance	Schedule anytime
Preperation	Draft, discuss, revise test protocol	Decide what you're going to show
What/ When do you test	Once, unless you have a big budget	Lots of small tests continually
Cost	£5k to £15k	£300
What happens after	20-page written report, takes a few weeks to arrive	Each observer takes one page of notes, development team debrief that day.

It is better to run multiple tests. This allows for main problems to be found in the first round of testing then smaller problems, which are equally as important, can be found in a subsequent round of testing. Overall, multiple tests will find more problems.

14.2.1 Usability room setup

This will preferably be a quiet room with a computer and a number of chairs. The participant will sit at the computer and perform tasks with the interface. The moderator sits over the shoulder of the participant and guides them through the process. In another room, the dev team observe either through a one-way mirror or webcam feed. With modern technology, it is now possible to record the test and watch back later.

14.2.2 Identifying Participants

Ideal participants will not know much about the app beforehand. Ideally you would want 3 or 4 users. The participants don't have to necessarily be the end user target group unless the app requires specific expert knowledge to use.

14.2.3 Facilitating a Study

Facilitators need to have decent people skills and be friendly. They need to tell users it is okay to make mistakes and encourage them to think out loud. Facilitators don't need to give hints to the users, however they should probe users for more details while taking notes without seeming too concerned with note taking. Its important they don't get upset if the user gets stuck, but uses this to get more information out of the user.

14.2.4 Types of Testing

Get It Test explores the users understanding of the sites basic purpose

Key Task test asks the users to do a specific thing and watch to see how they do it

Exploratory/ Formative explores the high level design concepts (can a user walk up and use it)

Assessment/ Summative explores lower level operations (can a user perform specific tasks)

Comparison tests explores differences between different prototypes or designs

Verification Test verifies that a UI is okay or that a fix works

14.2.5 Approach to Testing

Below is the outline of a usability test plan

1. Type of test
2. Purpose/ Goals/ Objectives
3. Participant Characteristics
4. Task List
5. Test environment/ equipment
6. Moderators Role
7. Evaluation Metrics and data to be collected
8. Report

This plan would be the kind of plan which would be used for a more traditional approach.

14.3 Conducting a Usability Test

Ideally the person observing the test should be out of view or out of the room. They should be looking for

- Does the use 'get it'?
- Can they find their way around the site?
- How long does it take them?
- Do they do any shocking things?
- What do the users like and dislike?
- What do they do when stuck?

14.3.1 Data to Gather

The observer writes down usability test notes. The development team will look over these notes and decide what to change. Development teams will generally value users actions and explanations over opinions and won't always listen to user suggestions for new features. It can be hard to figure out how to fix problems.

Small changes (tweaks) are often better than huge changes. Often removing or simplifying is better than adding, the first things to fix are the easy ones or will provide the highest reward.

To gather numerical data is very useful as it allows for direct comparison between different tests. Example data to collect could include the number or percentage of something; the count of something; or the time taken to do something.

14.3.2 Performance Goals

Some tests have specific goals, for example "every user will be able to find/ use the navigation bar". In these tests the moderator offers less guidance and fewer hints. There is less emphasis on thinking out loud. After the test, the users may be asked to replay steps and talk through them.

It is *bad* to set goals such as 'is the product usable', 'is it ready for release', 'is this a good product'. Goals should be narrow and set to a single thing rather than very broad.

Users will fail tests. They will often struggle to understand the point of the site; use different vocabulary; feel the site is too busy. However, it can be okay if the user is able to recover from the mistake and continue with the test.

14.3.3 Limitations

The tests are set in artificial environments. Some results won't prove that a UI 'works'. It may be better to use another UI evaluation method, for example a UI expert. It is also possible that a UI has an internal learning curve which the test doesn't allow for however when you get accustomed to the system it can be very powerful. Finally, it doesn't tell you if the market wants/ needs a product like yours.

Page 15

Designing for Accessibility

📅 2023-04-25

🕒 13:00

🎓 John

📍 RB LT1

15.1 What is Accessibility

Accessibility

Accessibility concerns removing the barriers that would otherwise exclude some people from using a service, product or system at all. It can be measured by the ease in which people with disabilities can perceive, understand, navigate and interact with the software technology.

There are a number of different types of accessibility which will be covered in this lecture:

Software Accessibility means that websites, apps, tools and technologies are designed and developed so that people with disabilities can use them.

Web Accessibility is a term used to identify the extent to which information on web pages can be successfully accessed by persons with disabilities including the aging¹.

The United Nations (UN) and World Wide Web Consortium (W3C) have published declarations and guidelines on ensuring that everyone can get access to information that is delivered through software technologies. Despite these standards existing, there is still a significant shortfall of compliance. This, in part, is the result of so many different people creating software technologies and now that using off-the-shelf frameworks such as *WordPress* are becoming more standardised, compliance levels are increasing. The shortfall in compliance prevents people with disabilities equal access of information.

A major challenge is to create a good user experience for people with disabilities that is both accessible and usable.

15.2 Impairments

Impairment

Something that has an adverse effect on people's ability to carry out normal day-to-day activities.

Impairments may prevent people from accessing web based information.

Limited Impairments

"Some people with conditions described would not consider themselves to have disabilities. They may, however, have limitations of sensory, physical or cognitive functioning, which can affect access to the system." (W3C, 2001)

¹(W3C, 2000)

Users are very diverse and for a website to be accessible to anyone it must be able to be accessed by any user regardless of their circumstances; which may include economics, location, communication infrastructure or a disability.

15.2.1 Types of Impairments

There are lots of different types of impairments.

Visual blindness, low vision, colour blindness

Motor cerebral palsy, Parkinson's disease, arthritis

Cognitive dyslexia, attention deficit disorder

Auditory sound, hearing impair audience

Speech stutter, speech impediment

15.3 Why is Accessibility Important?

Use of the internet and digital systems are spreading rapidly into all areas of society. It is becoming more important to have a connection to these systems and for those with impairments they may not be able to use them. Limited accessibility reduces the internet's potential as an effective tool.

Assistive/ adaptive software will not work if webpages are incorrectly coded.

The W3C have setup a working group (Web Accessibility Initiative - WAI) who work in conjunction with organisations around the world, and works to make the web more accessible. In 2004, an extension to the Disability Discrimination Act stated that all E-Commerce sites must be accessible.

15.4 Design for All

Design for All is an approach for designing for accessibility and is known as *universal design*. It goes beyond the design of user experience and applies to all user endeavours. The principles of universal design include

- Equitable Use
- Flexibility in use
- Simple, intuitive use
- Perceptible information
- Tolerance for error
- Low physical effort
- Size and space for approach and use

15.4.1 Inclusive Design

Designing for accessibility is based on four premises.

1. Varying ability is not a special condition of the few but a common characteristic of being human and we change physically and intellectually throughout our lives.
2. If a design works well for people with disabilities, it works better for everyone.

3. At any point in our lives, our self-esteem, identity and well-being are deeply affected by our ability to function in our physical surroundings with a sense of comfort, independence and control.
4. Usability and aesthetics are mutually compatible.

15.4.2 Web Accessibility

The *Web Content Accessibility Guidelines* (WCAG) can be categorized according to success criteria.

Perceivable users must be able to perceive the information being presented (it can't be invisible to all their senses).

Operable users must be able to operate the interface (the interface cannot require interaction that a user cannot perform)

Understandable user must be able to understand the information presented to them and the operation of the interface (the content of the operation cannot be beyond their understanding).

Robust users must be able to access the content as technologies advance (as technologies and user agents evolve, the content should remain accessible).

There are a number of Web Accessibility Checkers available online which can be used to check webpages for conformance to W3C standards.

Page 16

Standards

📅 2023-05-02

🕒 13:00

🎓 John

📍 RB LT1

16.1 Standards

A standard is an agreed way of doing something which has been created by a recognised body. They assure a level or quality or attainment and are specific and measurable. Standards are officially documented.

We need standards for a number of reasons:

Best Practice is promoted through the use of standards. This is important in areas, such as web usability, which are still young and contains many conflicting opinions on what makes a website usable.

Consistency is an important factor in creating websites that are simple to use. Where standards are followed correctly, you should be able to transfer from one organisation to another with relative ease.

Independent The guidance in standards does not represent the opinion of one company or one usability expert but presents a balanced, authoritative view.

Business Companies can ignore your research findings but they can't ignore standards since compliance is a mandatory requirement in many contracts (especially in the EU).

16.1.1 International Organisation for Standardisation

The International Organisation for Standardisation (ISO) gives world-class specifications for services and systems, to ensure quality, safety and efficiency. They have a wide range of standards for a wide range of applications.

They help companies to access new markets, level the playing field for developing countries and facilitate free and fair global trade.

16.2 Guidelines

A guideline is a general rule which is advisable to follow, a statement or policy or procedure, a piece of advice and a piece of evidence based practice. Guidelines can be *principles* (an abstract concept) or the steps required to achieve the principle.

16.2.1 Benefits of Guidelines

- Guidelines provide a clear instruction on a range of issues that designers will encounter
- Guidelines help usability specialists to evaluate the designs of their products
- It provides a good overview of the wide range of usability and Web design issues that designers may encounter when creating websites

- Applying the guidelines helps to reduce the negative impacts of 'opinion-driven' design and referring to evidence-based guidance can reduce the clashes resulting from differences of opinion between design team members.

16.3 Evaluation

Evaluation is the process by which the interface is tested against the needs and practices of the user. It is hoped that evaluation will get rid of any problems that might be presented in the system.

Evaluation allows user experience designers to learn about what users think and what makes a good system as well as finding out how effective and efficient the software being studied is; how much the users enjoy using it; how much it annoys and frustrates them; and where they get stuck.

16.3.1 Research Questions

There are seven key research questions for evaluation of usability:

Who is using the product?

What is it they need and are they comfortable with?

When are they using it?

Where are they using it?

Why are they using it?

How do people see a product in the context of what it is they are trying to accomplish in their life?

How many people are you creating the experience for or how many people do you need to test with?

16.3.2 Summative Evaluation

Summative evaluation is often undertaken at the end of the development process. It provides an evaluation or summary of the end product which allows the system to be matched to the requirements specification.

16.3.3 Formative Evaluation

Formative evaluations are undertaken during the development process. This means it doesn't have to have a complete build which can be used, a diagram or mock-up can be used.

Used to inform the development process which is why a prototype is the best thing to evaluate rather than a complete system. It takes account of users (knowledge, skills, gender, age, disability) and takes account of users tasks or goals.

16.3.4 Analytic Evaluation

Analytic evaluation consists of formal methods for analysing interfaces against Heuristic evaluation. This might be completed via a cognitive walk through or Goals Operations Methods Selection Rules (GOMS), both of which are task related. Aims to investigate existing situation, not envision new systems.

Goals Operations Methods Selection Rules (GOMS)

Goals is the task to do (eg send an email)

Operators are all the actions needed to achieve the goal (eg amount of mouse clicks to send email)

Methods are the group of operators (eg move mouse to send button, click the button)

Selection are the decisions which users have to make (eg 'move mouse to send button, click button' or 'move mouse to send button and press enter')

Cognitive Walkthrough

This is a technique used to evaluate the learnability of a system. Unlike user testing, it doesn't involve users (which means it can be relatively cheap to implement) as, like heuristic evaluations, it relies on experts to assess the interface.

At its core, a cognitive walkthrough has three parts:

1. Identify the user goal you want to examine
2. Identify the tasks you must complete to accomplish that goal
3. Document the experience while completing the tasks

The final stage asks questions: will users understand how to start the task; are the controls conspicuous; will users know the control is the correct one; was there feedback to indicate you completed (or did not complete) the task.

16.3.5 Empirical Evaluation

In this method, the users participate in trials of the prototype interfaces. It requires careful design of the trial's content and conduct which may involve benchmark tasks, may involve collecting and processing subjective opinions, and will involve evaluating with user participation.

Empirical evaluation may be done through field studies, lab-based usability evaluations or controlled experiments.

16.3.6 Lab-Based Usability Evaluation

Typically this contains representative tasks. Before the testing starts, the things to measure & observe have to be decided - these may be ease of learning or ease of performance. As this testing is conducted in a 'lab' setting, the participants of the tests may behave in different ways to the way they are expected to which can skew results. However, this should not discourage from using this method.