CBCS SCHEME

USN 1 C R 1 9 M C A 0 6

18MCA552

Fifth Semester MCA Degree Examination, Feb./Mar.2022
Principles of User Interface Design

Time: 3 hrs. Max. Marks: 100

Note: Answer any FIVE full questions, choosing ONE full question from each module.

Important Note : 1. On completing your answers, compulsorily draw diagonal cross lines on the remaining blank pages.
2. Any revealing of identification, appeal to evaluator and /or equations written eg, 42+8 = 50, will be treated as malpractice.

Module-1

1 a. What are the five measurable human factors are control to evaluation? Explain. **(10 Marks)**
b. Briefly explain the concept of motivations in human factors in design. **(10 Marks)**

OR

2 a. Explain the concepts of principles in PUID. **(10 Marks)**
b. Briefly explain the concepts of Goals of our profession. **(10 Marks)**

Module-2

3 a. Explain the different stage of LUCID. **(10 Marks)**
b. Explain the concept of Ethnographic observation. **(10 Marks)**

OR

4 a. Explain the scenario development in PUID. **(10 Marks)**
b. Explain the variety of expert-reviews methods. **(10 Marks)**

Module-3

5 a. Explain any five advantages of WYSIWYG word and processor. **(10 Marks)**
b. Explain OAI model explanation of direct manipulation. **(10 Marks)**

OR

6 a. Explain five challenges of programming in User Interface (PITUI). **(10 Marks)**
b. Explain Marcus applies semiotics as a guide to four levels of icons design. **(10 Marks)**

Module-4

7 a. Explain direct control pointing devices and comparisons of pointing devices. **(10 Marks)**
b. Explain Filt's law in pointing devices. **(10 Marks)**

OR

8 a. Explain limitations of short-term and working memory in theoretical foundation. **(10 Marks)**
b. What are the primary factors influences user's expectations and attitudes regarding response time? **(10 Marks)**

Module-5

9 a. Briefly explain the concept of reading from paper versus from display. **(10 Marks)**
b. Explain with a neat diagram, use of the OAI model to design manuals. **(10 Marks)**

OR

10 a. Which are the great attraction to making technical manuals available on the computer. **(10 Marks)**
b. Briefly explain the concept of online tutorials demonstrations and animations. **(10 Marks)**

* * * * *

1 a. What are the Five measurable human factors are control to evaluation?

Measurable human factors issues

Once a determination has been made of the user community and the benchmark set of tasks, then the human factors issues can be examined. Again and again I returned to these five measurable human factors issues: -

1. **time to learn.** How long does it take for typical members of the target community to learn how to use the task relevant set of commands.
2. **speed of performance.** How long does it take to carry out the benchmark set of task?
3. **rate of errors.** How many and what kind of errors are made in carrying out the benchmark set of tasks? Although time to make and correct errors might be incorporated into the speed of performance, error making is such a critical component of system usage that it deserves extensive study
4. **subjective satisfaction.** How well did users like using aspects of the system? This can be ascertained by interview or written surveys which include satisfaction scales and space for free form comments.
5. **retention over time.** How well do users maintain their knowledge after an hour, day, or week? Retention may be closely linked to ease of learning, frequency of use plays an important role.

1 b. MOTIVATIONS FOR HUMAN FACTORS IN DESIGN

The enormous interest in human factors of interactive systems arises from the complementary recognition of how poorly designed many current systems are and from the genuine desire to create elegant systems which effectively serve the users. This increased concern emanates from three primary sources: life-critical systems, industrial/commercial uses, and office, home, and entertainment applications.

Life-critical system s

Life-critical systems include air traffic, nuclear reactor, or power utility control, medical intensive care or surgery, manned spacecraft, police or fire dispatch, and military operations. In these applications high costs are expected, but they should yield high reliability. Lengthy training periods may be acceptable to obtain rapid, error free performance. Subjective satisfaction is less of an issue and retention is obtained by frequent use.

Industrial/commercial uses

Typical industrial commercial uses include banking insurance order entry, inventory management, airline, hotel, or car rental, utility billing, credit card management, and point-of-sales terminals. In these cases, costs shape many judgments; lower cost may be preferred even if there is some sacrifice in reliability. Operator training time is expensive, so ease of learning is important. The trade-offs for speed of performance and error rates are decided by the total cost over the system lifetime. Subjective satisfaction is of modest importance and again retention is obtained by frequent use.

2. a. Concepts of Principle of PUID

As the popularity the developed software application based on the user interface design of the software product so the user interface design principles are helpful to improve the quality of user interface of the software application. These principles mainly deal with Graphical User Interface (GUI) or command line interface of the software application. The user interface should be clear, meaningful, understandable in nature. There are multiple UI design principles.
Given below are user interface design principles:

- The Concept of Structural
- The Concept of Simplicity
- The Concept of Visibility
- The Concept of Feedback
- The Concept of Tolerance
- The Concept of Reusability

Let us study above principles in detail:

1. The Concept of Structural

This is the initial stage of to design the software application user interface which deals with overall UI architecture. It provides the structural data of the software application with combined the application related data and separated the unwanted things. This concept makes the software application to be well Organised (meaningful), Innovative (technological development), understandable (purpose of the software application) and useful (satisfy certain criteria).

2. The Concept of Simplicity

This process indicates the design of software application user interface should be simple, long lasting, user friendly with user control over user interface, sustainability, clarification on complexity, accurate throughput response, maximum usability and user's language. We expect our user do mistakes on UI, justify our data's, don't develop busy interface and explain the rules that are the best tips for the design of user interface for the software application. It will always helps to reduce the number of tasks to complete the total actions on the UI and protects the users data into the software application.

3. The Concept of Visibility

This concept of UI design technique involves the visibility of user interface for a software application. It mainly deals with graphical user interface i.e. the alignment is correct or not, spelling checking, position of logo or banners, consistency, easy to navigate with status, colour & brightness of UI and without hesitation to the user. So that the look & feel of the interface should be perfect, clarity, progressive disclose, transparency and error preventive. We should emphasis on the performance issue of the UI.

4. The Concept of Feedback

This principle indicates the enhancement of user interface view depends of feedback. It always welcomes to change or modify the UI depends upon the user or client reviews. So the designer always analyse the process, view, any ambiguity issues, bugs, technological advancement, change of conditions and to make more user friendly of user interface of the software product. For users every action on the UI returns a meaningful and clear reaction or feedback.

5. The Concept of Tolerance

This is one of the concepts of user interface design for the software application which deals or affects the budgeting of the application. The UI designer will design the interface that will be always flexible and tolerant, so it reducing the cost of rework of user interfaces change. For any stage of the application the UI should be tolerable in sense.

6. The Concept of Reusability

This is the last technique of the user interface design. The designer should design the way where the user interface view should be reusable inside the software application. Some internal or external components, behaviour of components, consistency of application should be reuse inside it. So that the user should not remember the interrelated process flow of the application on the user interface. It provides some short cut keys to the user for easy to work on that.

2.b. Goals for our profession

- Potential research topics
  - Reducing anxiety and fear of computer usage
  - Graceful Evolution
  - Specification and implementation of interaction
  - Direct manipulation
  - Input devices
  - Online assistance
  - Information exploration
- Providing tools, techniques, and knowledge for system implementers
  - Rapid prototyping is easy when using contemporary tools
  - Use general or self-determined guideline documents written for specific audiences
  - To refine systems, use feedback from individual or groups of users
- Raising the computer consciousness of the general public
  - Many novice users are fearful due to experience with poor product design,
  - Good designs help novices through these fears by being clear, competent, and nonthreatening

3a. Stages of LUCID

LUCID		
Logical	**User Centered**	**Interaction Design**
The design process builds on a strong conceptual model. Iterative review and refinement includes user feedback at all critical stages. Successive prototypes and team reviews allow opportunities for technical review and ensure viability of the design	Software is designed in the context of the overall tasks and work flow (including both manual and computerized activities). Design is based on user activity and employs the user's language and context. The design model fits the user's mental model rather than the technical implementation model.	Interaction design is treated as distinct from technical design. The scope of the design is "everything but code" and includes: ❑ look and feel ❑ language ❑ screen objects & layout ❑ navigation ❑ user assistance

LUCID is organized into six stages:

Stage 1: **Envision**	Develop UI Roadmap which defines the product concept, rationale, constraints and design objectives.
Stage 2: **Analyze**	Analyze the user needs and develop requirements.
Stage 3: **Design**	Create a design concept and implement a key screen prototype.
Stage 4: **Refine**	Test the prototype for design problems and iteratively refine and expand the design.
Stage 5: **Implement**	Support implementation of the product making late stage design changes where required. Develop user support components.
Stage 6: **Support**	Provide roll-out support as the product is deployed and gather data for next version.

3b. Concepts of Ethnography

Ethnography is a study through direct observation of users in their natural environment rather than in a lab. The objective of this type of research is to gain insights into how users interact with things in their natural environment.

Ethnography is a qualitative research study looking at the social interaction of users in a given environment. This research provides an in-depth insight into the user's views and actions along with the sights and sounds they encounter during their day. It provides the researcher with an understanding of how those users see the world and how they interact with everything around them.

Ethnography methods include direct observation, diary studies, video recordings, photography and artefact analysis such as devices that a person uses throughout the day. Observations can be made anywhere from the user's workplace, their home or while they are out with family and friends. The length of the studies can vary depending on the research that is being conducted. They can range from a couple of hours of observation, to studies that last several months.

There are two methods for observation:

1. Passive observation

Passive observation which can also be known as 'shadowing' is where a user or users are shadowed while they go about their everyday tasks observed by a researcher. Sometimes before the research begins, users will be interviewed on their own or in groups to learn more about them and their needs. Observations will be documented throughout the day using a number of methods such as taking notes, photographs, sketches or videos. The research may be conducted as part of a team so that a larger number of users can be observed and therefore gaining a greater insight quickly. This observation method is a good way for researchers to see how users go about their day first hand and identify any disconnections of when the user tells the researcher one thing but actually interacts in another way.

2. Contextual interviews

Contextual interviews are where the researcher will interact with users while observing them going about their everyday tasks. The interviews will be held in a natural environment, so as not to feel too formal. The researcher will observe the user going about their everyday tasks and ask questions to gain insight.

Analysis

The analysis of the findings will vary depending on the method that has been used to gather the insights. In both cases though, the research was to obtain an in-depth view of the users and how they go about completing tasks which are under review. Researchers will look for patterns and themes from the data. They will look for the challenges and barriers that users encountered and how this effected different users.
One of the methods used to analyse the data is the use of an affinity diagram. This method allows you to take all the observations from the research and group them together so that

you can begin looking for patterns. Researchers will put all their key points onto Post-It notes and then categorise these into groups which relate to the same topic.

Advantages of Ethnography research

- Ability to see first-hand how users interact with technology in their natural environment
- Identify unexpected issues that you might not have encountered in a usability test
- Opportunity to test new product ideas before they are released to the market to see what demand is like

Disadvantages of Ethnography research

- Because there is a greater insight into the user it takes much longer to generate and analyse all the findings.
- Short studies may not get a user acting naturally as they are aware of the researchers present.
- The cost of conducting ethnographic studies is typically much higher than conducting a usability test.

Conclusion

Ethnographic studies are a good way to really understand your users and the challenges they may face while going about their everyday lives. The research will give you insights to your users that you may not have seen if they were in a lab being asked to complete a task. However, Ethnographic studies can be costly and time-consuming, so making sure that you get the research method right is crucial to making sure that you are getting the research questions answered. Having conducted a study, you then need to present your findings back in an informative and meaningful way that will allow teams to use the information to make informed changes, making sure that your own opinions have not come into the findings.

4 a. Concept of Scenario Development in PUID

Design scenarios are useful tools for communicating ideas about user actions. Mapping design scenarios also has the added benefit that it helps formalize ideas and to take creative approaches to those ideas. Most importantly of all, it will ensure that your designs are firmly rooted on terra firma and taking a “what our users want/need” approach from the outset. In general, user scenarios are designed to capture the key interactions with a system and not all possible interactions.

Scenario mapping can be effective at many points during a project but there are three great uses for it at specific stages:

- Ideation – if you're trying to create a new product, then having scenario maps makes it very easy to explore ideas with your team and with users. It also helps, in a similar means to task analysis, formulate a shared vision for the project. (See the method below as to how to do this).
- Iteration – if you're new to a product and you're going to be involved in creating future iterations then it's pretty easy to create scenario maps "on the fly" by observing users with the current product. (This can be done solo and doesn't need the method below).
- Usability testing – user scenarios can also be used to define which are the most important areas to test during usability testing and to provide guidance on how it should be done.

How to Do User Scenario Mapping?

It's a pretty simple process to create user scenarios and here's a simple way to conduct it:

- First, find a place that you can use to get creative – you're going to need somewhere where a group can talk and discuss without being interrupted and without disturbing others. You're probably going to need between 2 and 3 hours for the session.
- Then invite a bunch of relevant people to the session – the UX team, the development team, the product manager, etc. but don't invite too many people, a maximum of 7 is a good idea as it means everyone can contribute without anyone getting lost in the mix.
- Then get some post-it notes, flipchart paper, etc. stuff that makes it easy to capture an idea and get it in front of everyone. Bring sellotape or blu-tack in case things aren't sticky enough for the surfaces in your room.
- Explain to everyone present what your objectives are and what a user scenario is – it's always good to have everyone on the same page. However, don't spend too much time on this either; you want people firing on their creative best not snoring in the corner because you've TMI'd them to death.
- Hopefully you have user personas because they'll come in handy as you map your first user scenario. What is it that this user must do in this interaction with the product? This tells you what goes into the scenario.
- You also need to provide context to make your scenarios as accurate as possible – the who, what, when, where and why detail that gives a scenario colour and makes it easy to relate to.
- Then it's time to take some baby steps and walk through the scenario in the shoes of your user (referring to your user personas). What will the user do? What information do they need to get that done? What questions will they need answered or will you need answered to do this? What assumptions will you have to make to make this work?
- Finally, you also want to collect ideas from the team that don't fit within the scenario but may be related to it.
- After you've completed each scenario make a written note of it and stick it to the wall. Try and get scenarios grouped so that you can make easy sense of them and spot any gaps that arise.

4b. Concept of Expert Review

An expert review or website audit is an inspection method designed to identify usability problems in an online product or service. The review is carried out by a small group of usability experts (between 1 and 4), who analyse the product or service to identify any potential usability issues.

Expert Review is a method wherein one or more UX experts act as users and test a design, application, or product to identify usability hurdles and make recommendations to improve them. In this article, I will be talking about two methods for conducting expert review.

- **Persona Testing** — A task evaluation method in which a UX expert performs tasks on behalf of the persona/user on an application, and measures success or failure of the task based on a pre-defined criterion. After completion of each task, the expert would list all the usability hurdles and suggest recommendations for improvement. The report is then discussed and shared with the product team. You do this exercise over a period of time to review the application before you take it to the user for testing. You can also analyze trend to see how the usability is improving.
- **Cognitive Walkthrough** — A scenario evaluation method in which multiple UX experts would perform a list of activities for each scenario the user is expected to encounter in an application. **The expert while performing the activities would ask these questions:**

Will the user try and achieve the right outcome?
Will the user notice that the correct action is available to them?
Will the user associate the correct action with the outcome they expect to achieve?
If the correct action is performed; will the user see that progress is being made towards their intended outcome?

Each expert identifies and lists all the usability hurdles in their own words with their recommendations. When the process is complete, all the findings are collated and shared as a single report with the product team and then the issues are prioritized for fixing. This method allows a holistic review of the application as every scenario encountered by the user is taken into consideration.

How to conduct Persona Testing

Example of Tasks
Add a new cost center.
View list of available job templates.
Organize and manage org locations.
Terminate existing employees.
Example of scenario
Sue Ivory has been working with GreenFields Inc. for the last 2 years as a design engineer. She is recently promoted and is eligible for company broadband benefit. She wants to view her pay stub to see how the additional benefit reflects in her salary.

- For each application, define the user personas such as Manager, Employee, Administrator, etc.
- List all the tasks the user is expected to perform.
- List the steps for each task.
- Define the measurement criteria.
- Perform the steps for each task on the application and note usability hurdles.
- After completing the task, give a rating based on the defined criteria.

Usability Rating Scale	
Fail - 1	Cannot complete the task.
2	Can complete the task, but with great difficulty.
3	Can complete the task, but flow is not logical.
4	Can complete the task with minimal errors or effort.
Pass - 5	Easily complete the task.

How to conduct Cognitive Walkthrough

- For each application, define the user personas such as Manager, Employee, Administrator, etc.
- List all the scenarios the user is expected to encounter while using the application.
- List all the activities for each scenario.
- Define the acceptance criteria.
- Perform the activities on the application and note usability hurdles.
- Repeat this exercise with 2–3 experts.

These methods are cost effective, quick, and can be performed by anyone in the UX team. Use these methods to test the application for usability from a user's perspective and provide quick feedback to enable decision making in the design process.

5a. Advantages of WYSIWYG Word Processor

The Advantages ~

- Anyone can edit.
- Wikis are easy to use and learn.
- Wikis are instantaneous so there is no need to wait for a publisher to create a new edition or update information
- A rich text editor offers a what-you-see-is-what-you-get (WYSIWYG) interface, facilitating the formatting and layout of content.
- People located in different parts of the world can work on the same content.
- The wiki software keeps track of every edit made and it's a simple process to revert back to a previous version of a page.
- Non-technical users have access to the power of web publishing.
- The wiki has no predetermined structure – consequently it is a flexible tool which can be used for a wide range of applications.
- There are a wide range of open source software wiki's to choose from so licensing costs shouldn't be a barrier to installing an institutional wiki.
- Anyone can create web sites and put them online
- Create web sites quickly
- No prior programming knowledge required – no need to know what HTML code looks like
- Provide a platform to start learn HTML
- Easy
- Fast
- No special skills required.

~ The Disadvantages ~

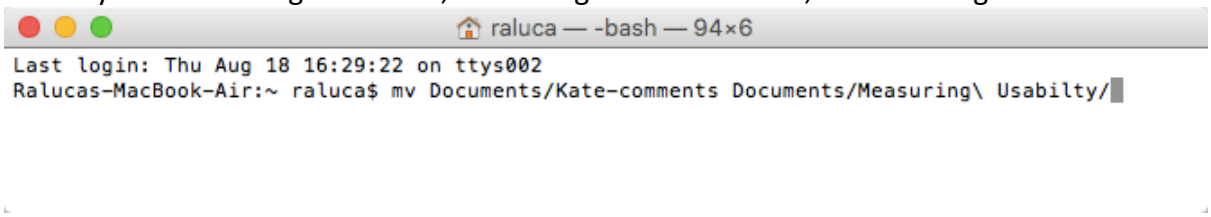
- Anyone can edit so this may be too open for some applications, for example confidential documentation. However it is possible to regulate user access.
- Open to SPAM and vandalism if not managed properly. There are easy ways to restore a page and in the case of WikiEducator you must be logged in to edit pages so this reduces vandalism by automated spam bots.
- Requires Internet connectivity to collaborate, but technologies to produce print versions of articles are improving.
- The flexibility of a wiki's structure can mean that information becomes disorganised. As a wiki grows, the community must plan and administer the structure collaboratively.
- Messy code
- Depreciated code
- potentially arbitrary and redundant code (dependent on the editor)

5b. OAI for Direct Manipulation

OAI model. The OAI model graphically represents the users' workplace using metaphors and let the users perform action(s) on the object. The sequence of work is to first select the object graphically (using mouse or other pointing device), and then performing an action on the selected object.
Shneiderman identified several attributes of this interaction style that make it superior to command-line interfaces:

- **Continuous representation of the object of interest**. Users can see visual representations of the objects that they can interact with. As soon as they perform an action, they can see its effects on the state of the system. For example, when moving a file using drag-and-drop, users can see the initial file displayed in the source folder, select it, and, as soon as the action was completed, they can see it disappear from the source and appear in the destination — an immediate confirmation that their action had the intended result. Thus, direct-manipulation UIs satisfy, by definition, the first usability heuristic: the visibility of the system status. In contrast, in a command-line interface, users usually must explicitly check that their actions had indeed the intended result (for example, by listing the content of the destination directory).
- **Physical actions instead of complex syntax**. Actions are invoked physically via clicks, button presses, menu selections, and touch gestures. In the move-file example, drag-and-drop has a direct analog in the real world, so this implementation for the move action has the right signifiers and can be easily learned and remembered. In contrast, the command-line interface requires users to recall not only the name of the command ("mv"), but also the names of the objects involved (files and paths to the source and destination folders). Thus, unlike DM interfaces, command-line interfaces are based on recall instead of recognition and violate an important usability heuristic.
- **Continuous feedback and reversible, incremental actions**. Because of the visibility of the system state, it's easy to validate that each action caused the right result. Thus, when users make mistakes, they can see right away the cause of the mistake and they should be able to easily undo it. In contrast, with command-line interfaces, one single user command may have multiple components that can cause the error. For instance, in the example below, the name of the destination folder contains a typo "Measuring Usablty"

instead of “Measuring Usability”. The system simply assumed that the file name should be changed to “Measuring Usablty”. If users check the destination folder, they will discover that there was a problem, but will have no way of knowing what caused it: did they use the wrong command, the wrong source filename, or the wrong destination?



The command contains a typo in the destination name. Users have no way of identifying this error and must do detective work to understand what went wrong.

This type of problem is familiar to everyone who has written a computer program. Finding a bug when there are variety of potential causes often takes more time than actually producing the code.

- **Rapid learning.** Because the objects of interest and the potential actions in the system are visually represented, users can use recognition instead of recall to see what they could do and select an operation most likely to fulfill their goal. They don’t have to learn and remember complex syntax. Thus, although direct-manipulation interfaces may require some initial adjustment, the learning required is likely to be less substantial.

6a. Challenges of Programming in UI

Everyone else can split hair all they want, and they may be right. Perhaps I’m generalizing. But *UI design* is *no different than web design*, at least not in the Design-Utopia of my dreams. The UI ‘niche’ (which used to be strictly software/electronic products domain) has extended way beyond those confines. And that is as it should be. To get some focus and limelight, you have to direct it all at the users first.While UI designers continue to bask in greater scope and opportunities of this incredibly amazing wave in Industry 4.0, there are some hidden challenges that can take all the fun right out of it. I’m not talking design patterns or reminding you to A/B split *test* it all. I am talking challenges so seemingly trivial they often escape notice, but niggle at you till the project end and beyond.

1. The New, ‘Rad Looking’ GUI Elements

There’s no end to new trends in designing Links, Buttons, Scrollbars, etc. for the web. We have ‘Hamburger’ menu icons, horizontal scrolling, ghost buttons, and more. While they look and feel amazing, you have to take a moment to ask yourself: “Are they making users’ lives difficult?”If the answer happens to be ‘yes’, then congratulations dear UI designer: you have just lost yourselves some users.

As someone who is paid to keep users and *usability* in mind at all times, UI designers have to ensure their products are usable, accessible, and pretty: **in that order**. Buttons are for immediate action and should be clearly visible. Horizontal scrolling may be used in specific context, but with the unavoidable side-effect of confusing the heck out of users. So be warned.

I hate to quote an unconscionably evil character, but Dolores Umbridge's words in this context are absolutely appropriate. "*Progress for progress's sake must be discouraged*". By all means, be trendy, but don't suddenly become Gwyneth Paltrow and lose all sense of reality in the progress.

2. Looking Like a Link (and Not Being One)

This is related to the first point up there.

We know users have little-to-no time or patience for wacky antics, even if they're looking for cat pictures and other silly memes. Even if your website is solely based on prank-products, puns, and jokes, *do not prank your own users with design*.

This should be a no-brainer, really.

We'll often come across websites with text that looks like a link. We take the cursor on it and it changes slightly too, of course it's a link. We click on it and… nothing. Absolutely nothing happens. *Why did it pretend to be a link if it isn't one*?

That's a trick question. The very act of **making unclickable text look like links is a trick**. There may be some contexts where its use won't irritate a majority of your users, but I honestly don't know what they might be.

3. Mythbusters: *Navigation* 'Must-Haves'

Given enough time, the usage and necessity of *anything* will be blown wildly out of proportion.

Cases in point: Hamburger menu and infinite scrolling. Both patterns are popular and used with abandon by scores of websites; *whether their information architecture needs it or not*.

We see infinite scroll used by some very popular platforms: Google Images, Facebook, Twitter, even e-commerce websites, and we think it's the solution to waning user interest and dumb *navigation*.

But ask yourself this: How will your users locate/ mark content? **How do they move back and forth from content they like? How will they cope with the sheer amount of information you're throwing at them?**

Then there's Hamburger menu: a magical icon that makes *navigation* troubles disappear! It's easy to see why you may be tempted (or already working with) a hamburger menu. *It's progressive disclosure, man*.

Hamburger menu is the design equivalent of stuffing all your furniture in the attic to make your home minimalistic.

Let me present the other side of that coin to you.

Users tend to skim through the websites. Whatever's apparent to them in those few precious seconds gets (somewhat) absorbed, while the rest can sit in the dark and gather for all they care. Hamburger menu is the design equivalent of you stuffing all your furniture in the attic to make your home minimalistic. NBC News tried and failed, so maybe try and learn from their mistakes. **Instead of pushing everything out of sight, make your menu visible with *tabs*: 4-5 is the sweet spot**. And if you can't compress everything into just a handful of categories, you need to take another look at your content and hierarchy.

4. May the Forms be with You

Any conversation about *UI design* is incomplete without mentioning forms. But don't worry. I won't be harping about inline validation (do) or progress bar (do) or form resets (don't) or placeholder text (depends on your audience).

All the basics, (and really they should be form basics by now) aside, I want talk about tone.

I've said this before: **forms should use conversational tones and focus on 'what users need to provide' instead of 'what you need from users**'. Ask a question in place of form field label.

A simple change shifts the focus from you to users, simplifies, and 'humanizes' a dead-boring and tedious task.

5. Clash with the Clients

This looks like a non-issue until you find yourself seriously questioning why on earth you decided to be a UI designer. It happens most often when clients and stakeholders are dismissing your ideas and squashing in their own.
Now, the more egoistic, aggressive designer will cuss and whip up a rage-storm before reminding The Man that the client(s) hired him/her and not the other way around. The calm, professional one will try to see things from clients' perspective before listing off alternatives and logical argument.
But sometimes, despite all your experience and best efforts, your design (and you) will still get rejected because communication failed between you and the clients. In those cases, accept defeat and move on.
In Essence
Challenges for UI designers can come from the most unexpected sources. I hope I left you 5-point wiser than before in this regard. Here's the *TL; DR* version:
Appearance of GUI elements: Standard is safe (and tested and better in general).
Don't tease your users with elements that appear to be GUI elements (but aren't).
Consider your information architecture before blindly going for patterns like hamburger menu and infinite scroll.
A conversational tone in forms can lessen the pain considerably.
Communicate with clients and understand their motivations as best as you can.

6b. Semiotics and Iconic Guidelines

SEMIOTICS IN USABILITY: GUIDELINES FOR THE DEVELOPMENT OF ICON METAPHORS

Semiotics is the study of signs and describes the perception, comprehension and communication of symbols. In the field of computer science, semiotics is especially relevant to the design of icons. A perfect metaphor is the prerequisite to unmistakably associate a function with an image.
All these guides primarily serve the creation of a consistent look-and-feel. But an icon rarely stands for itself. Therefore the integration into an icon set needs to be considered, whereby homogeneity (the icons of a set should belong together) and heterogeneity (within a set the risk of mix-ups should be minimized) become relevant.
Along the aspects of homogeneity, heterogeneity and a graphically appropriate design, the most important premise for unambiguous symbols is the distinct association between the underlying function and its visual depiction. This relation is called a metaphor.
When creating a new set of icons the metaphors need to be defined first. They have to be a representable of the function within the whole functionality, easy to recognize and understandable. The results of a study on LibreOffice icons will illustrate most important aspects how to find these metaphors.

1. Apply metaphors only once.

On the one hand an icon set should be homogeneous and show the affiliation between the items. On the other hand there should be sufficient discrimination between the elements of a set to avoid mix-ups. This problem is especially relevant if the same metaphor is used for different functions. In our LibreOffice study we found this problem in the functions *Format Paintbrush* and *Show Draw Functions*: Both use a brush as metaphor.

2. Rethink conventionally used metaphors.

A lot of metaphors have been iconized so much that they were integrated into common usage. For more than 50 years a green triangle is used for play, a gray square for stop and a red circle for record in music players (and respectively software). A little more recent, but similarly famous are some functions we investigated in the context of word processing. But in our detailed analysis of *Copy/Paste* and *Redo/Undo* it becomes apparent that the usual depiction of these icons has a tendency for mutual mix-ups.

Even though it is obvious that things are not working well, changes in those quasi standardized solutions should only be done with great care. However, we received a lot of interesting suggestions how to trigger these problems in the comments to our original articles. For example, the short-cuts ctrl+c and ctrl+v are so well-known that an icon might be created on the grounds of this information.

3. Antiquated metaphors might work well.

At the same time not all conventionally used metaphors need an overhaul, even when the underlying functions are not used in the same way anymore. In our study this was apparent for *Save*. This depiction of a floppy disc is intuitively associated with its function and clearly exceeds the alternative, even though a lot of people actually never touched a floppy disc.

4. Adjust the degree of abstractness according to familiarity of the metaphor.

While creating a metaphor the degree of abstraction from the original is of utmost importance. A lightning bolt can stand for 'fast' or 'immediate', but at the same time it can be interpreted as energy or hazard. The guiding principle should be deemed: The better known a function, the more abstract the metaphor can be. However, less known functions need explicit metaphors and a figurative support. In LibreOffice there are a few seldom used functions (e.g. the *Navigator*, that should be placed less prominently or require a more pictorial icon.

5. Arrows are not specific.

Human action and every software procedure has an immanent sequence. It is an obvious choice to use this sequence for the depiction of the icon; for example to use a downward pointed arrow to represent the storing of data from working memory into a file (see figure 2). You can clearly see the preference of designers for arrows in the comments to our detailed articles: There were a lot of proposed solutions that apply arrows. Next to the unspecific overflow of arrows, they might be problematic in sinistrograde localizations and when the sequence is ambiguous. Hence, when searching for a metaphor, try to avoid arrows as much as possible.

6. Define metaphors independent from language and culture.

Upon creating an icon, it should be considered that the original label of a function might not be understood all over the world. A well-known example is the term 'Burn CD' which is translated in French as 'engrave' ('Graveur CD'). Therefore icons representing this term should not be based on the metaphor of fire. Similarly, colors can be understood differently and even pictograms are not necessarily standardized.

7. Make icons simple.

An icon should be rapidly recognized by its symbolic depiction and not be mixed-up with other icons. The simpler an illustration, the better these requirements are fulfilled. Two ways lead to these easy solutions: a perfect metaphor, like the scissor for the function *Cut* or a branding, respectively the establishment of a distinct symbolism, e.g. the logo of the Adobe Acrobat for the output of PDF documents. These icons get optimum values in the test.

7a. Direct Control Pointing Devices

Direct devices have no intermediary; the movement of the body equals the input to the machine. Examples of direct devices are **touch screens, light pens, and voice recognition systems**.

Pointing devices can be grouped into :

i. Direct control on the screen surface, such as the touchscreen or stylus, and

ii. Indirect control away from the screen surface, such as the mouse, trackball, joystick, graphics tablet, or touchpad.

Direct control pointing devices :

1. **Lightpen :** The lightpen is a device that enabled users to point to a spot on a screen and then press a button to perform a select, position, or other task.
2. **Touchscreen :** Touchscreen is robust and does not require picking up an external device; instead, it allows users to make direct-control touches on the screen with a finger.
3. **Tablet PCs and mobile devices :** Tablet PCs and mobile devices make it natural to point on the LCD surface, which can be held in the arm or hand, placed on a desk, or rested on the lap.
4. **Stylus :** The stylus is an attractive device because it is familiar and comfortable for users, and users can guide the stylus tip to the desired location while keeping the whole context in view.

Indirect control pointing device :

1. **Mouse :** The mouse is appealing because of its low cost and wide availability. The hand rests in a comfortable position, buttons on the mouse are easy to press, long motions can be done rapidly by moving the forearm, and positioning can be done precisely with small finger movements.
2. **Trackball :** Trackball is a movable ball that mounted on top of the stationary device. The ball gives an advantage for good locations where a mouse cannot move around enough. It is widely use in design and manufacturing industries.
3. **Touchpad :** Touchpad are slightly same as touchscreen, but touchpad can be only use in their provided space, commonly square or rectangle in shape. It works by sliding finger over their small flat surface, and tapping the finger on the surface to click an option. It widely used in laptops.
4. **Pointing stick :** Pointing stick is a pressure sensitive small nub used like a joystick. It is located between the keys on a laptop keyboard that allows the user to control the pointer by directing the stick with one finger. Commonly found in between 'G' and 'H' key, and also at below the spacebar.

7b.Fitts Law

Vertical Orientation

In Josh Clark's "Designing For Touch" article he also points out that most people hold their tablets at the top of the device. Therefore the top two corners become the most important zones to place actionable targets. In my experience, I've seen many people hold it at the bottom. This is especially true for typing when users have the iPad's split keyboard turned on. In this case the bottom corners would be the most important zones, except when typing is **needed.**

Horizontal Orientation

Tablets can be held by one hand vertically with relative ease, but horizontally it is a bit more of a challenge. The bulk can increase the proclivity to drop the device. Therefore if tablets are being held in a horizontal orientation it is almost always a two-handed operation. We also need to keep in mind that one side typically has a home or menu buttons on the device so that side has even less space for the thumb to operate on the screen.

"Docking" Mode

All the above is dependent upon tablets being held, but because of the tablet's size and bulk it is often in a "docked" mode. This is not to say that it's plugged in but that it's often placed on our laps, desks or tables. The new iPad case that folds back to create a small stand makes this even more true. This allows for more pointing on the tablet than on the smartphone. Therefore design is not as restricted to the thumb zones and expands our original Fitts' Law equation to the entire touch surface.

Seamless Switching

It's important to remember that switching modes and orientations happens often for users. Users will switch modes without even thinking, going to the most convenient for the activity they are performing in the moment. Now, there is some delay time for going between vertical and horizontal so that switch is not quite as seamless. The point remains that we certainly need to design with the points above in mind, but it's not as if users are locked into any particular mode or orientation. We need to be cognizant of the mode and orientation the user will likely be in for a given activity/action and design with that in mind.

Orientation and how the device is held or stationed becomes critically important in our designs. Fitts' Law can be affected in various manners and in order to design for optimum UX it's important to know how. Fitts' Law is certainly not the only, or even most important design consideration, but it's almost always a good starting point. There are certain challenges we face on mobile that we don't have with the desktop. With mobile growing as fast as it is, these considerations should be a standard weapon in the UX repertoire.

8a. Limitation of Short Term Memory

- While long-term memory has a seemingly unlimited capacity that lasts years, short-term memory is relatively brief and limited. Short-term memory is limited in both capacity and duration. In order for a memory to be retained, it needs to be transferred from short-term stores into long-term memory.

- Short-term memory, also known as primary or active memory, is the capacity to store a small amount of information in the mind and keep it readily available for a short period of time.

- Most of the information kept in short-term memory will be stored for approximately 20 to 30 seconds, but it can be just seconds if rehearsal or active maintenance of the information is prevented.

- The classic model, known as the Atkinson-Shiffrin model or multi-modal model, suggested that all short-term memories were automatically placed in long-term memory after a certain amount of time

- Short-term memory has two major limitations; the first is that you can only store a small amount of information, and the second is that the memory decays over time.

- Most adults can store between 5 and 9 items in their short-term memory. This idea was put forward by Miller (1956) and he called it the magic number 7. He though that short term memory could hold 7 (plus or minus 2 items) because it only had a certain number of "slots" in which items could be stored.

8 b. User Expectations and Attitudes regarding Response Times

Response times (RTs) are a natural kind of data to investigate cognitive processes underlying cognitive test performance. We give an overview of modeling approaches and of findings obtained with these approaches. Four types of models are discussed: response time models (RT as the sole dependent variable), joint models (RT together with other variables as dependent variable), local dependency models (with remaining dependencies between RT and accuracy), and response time as covariate models (RT as independent variable). The evidence from these approaches is often not very informative about the specific kind of processes (other than problem solving, information accumulation, and rapid guessing), but the findings do suggest dual processing: automated processing (e.g., knowledge retrieval) vs. controlled processing (e.g., sequential reasoning steps), and alternative explanations for the same results exist. While it seems well-possible to differentiate rapid guessing from normal problem solving (which can be based on automated or controlled processing), further decompositions of response times are rarely made, although possible based on some of model approaches.

Cognitive tests are meant to measure abilities. Abilities refer to levels of performance, whereas processes are the activities involved in reaching a performance outcome. Typically, cognitive tests do not yield process measures. It is perfectly possible to measure an ability without knowledge of the processes that are involved, but then the resulting measure only describes the level of performance, which is not always satisfying because it leaves why questions unanswered. Explanation requires a narrative of how something comes about. Processes provide such a narrative. Processes do not only help for understanding, they also help for more informative feedback and knowing the processes may help for interventions and remediation. Process information is also relevant to make validity inferences in the positive sense if the inferred processes support the interpretation of the intended ability, and in the negative sense, for example, because unintended processes can invalidate a measurement result. An important example of an invalidating process is guessing. Like it is possible to measure without investigating processes, it is also possible to investigate processes without measuring the related abilities, and a combination of the two is also possible.

Processes have the intrinsic feature that they take time. Therefore, response times are natural and evident kinds of data to investigate processes. Other kinds of data can also be informative regarding processes involved in reaching or not reaching a certain performance level. In fact, the responses themselves may be informative. For example, based on a cognitive theory stipulating the processes involved in finding the correct response to a set of test items, a model can be developed for the probability of a correct response based on the mastery of the process skills required to successfully respond to the items. This is the basic principle behind cognitive diagnostic modeling (Rupp et al., 2010). Mediation research can also contribute to process research because the mediation variable functions as a process in the narrative of how the level of a dependent variable comes about (Hayes, 2017). It may explain why mediation analysis has become so popular. As far as types of data are concerned, eye movement data are an interesting source of information regarding processes (Cho et al., 2018), because it may be assumed that the mind follows the eyes, or the eyes fixate the

stimuli the viewer is processing. Furthermore, brain activation and EEG data can be useful, as well as actions such as clicking and moving on the computer screen to find an answer to a question.

Here we will focus on response times, the time a respondent takes to respond to individual items in a cognitive test. Making use of response times in modeling test data can lead to the identification and measurement of processes, but, as will be discussed, the use of response time information does not necessarily imply it leads to inferences regarding the processes which are involved. The scope of this article comprises modeling approaches in which response times are used and cognitive process inferences can be made. For more general reviews of the use and importance of response time and of time available to make a test, see reviews by Lee and Chen (2011); Kyllonen and Zu (2016) and Schnipke and Scrams (2002).

Response time modeling approaches can be classified into four very broad possibly overlapping and not necessarily homogeneous categories. The categories are partly inspired by an overview made by van der Linden (2009). Before listing the categories, we introduce a symbolic notation for the models:

T_{pi} for the response time of person p and item i;

A_{pi} for the response accuracy of person p and item i;

$\leftarrow$ to indicate which variable is the dependent or independent variable; for example, $T_{pi} \leftarrow$ means that response time is the dependent variable.

(a) *Response time models*: response times as the sole end variable ($T_{pi} \leftarrow$);

(b) *Joint models*: response times as one of the end variables, jointly with another kind of variable (e.g., accuracy) ($[T_{pi}, A_{pi}] \leftarrow$);

(c) *Dependency models*: joint models in which response times and other data (e.g., response accuracy) are jointly modeled with the possibility of dependencies beyond dependencies captured by latent variables and item parameters ($[T_{pi} \leftrightarrow A_{pi}] \leftarrow$);

(d) *Response times as covariate models*: response times as an origin variable and another kind of variable (e.g., accuracy) as the end variable ($A_{pi} \leftarrow T_{pi}$).

An *end variable* is an outcome variable, also called dependent variable, the last variable in a dependency network. For example, in a simple measurement model for speed, the observed response times are modeled as a function of a latent speed variable and item time parameters. More than one variable can have the status of an end variable. For example, response time and response accuracy (correct vs. incorrect) can be joint end variables. An *origin variable* is a covariate, also called independent variable, a variable in the dependency network that is not explained by any other variable. More than one variable can have the status of origin variable.

9a. Reading from Paper and Display

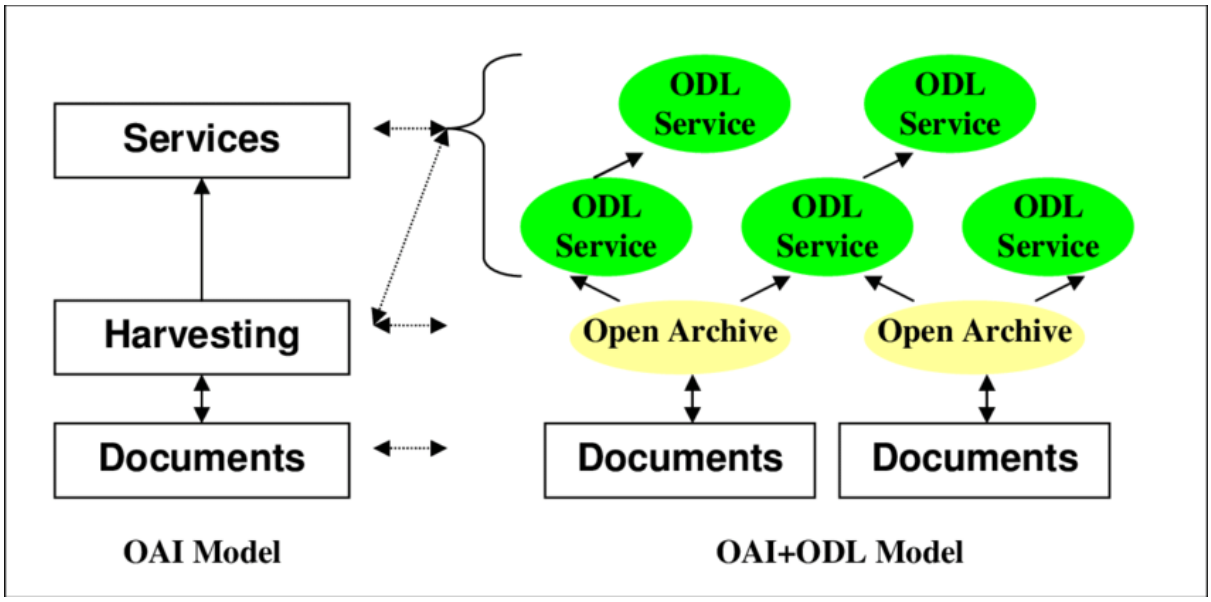
- When it comes to learning about computer systems many people experience anxiety, frustration, and disappointment
- Even though increasing attention is being paid to improving interface design, complex systems can still benefit both paper and online help
- Forms of paper user manuals:
  - Install manual
  - Brief getting-started notes
  - Introductory tutorial

  - Thorough tutorial
  - Detailed reference manual
  - Quick reference card
  - Conversation manual
- There are many reasons to have online manuals
  - Physical advantages
  - Navigation features
  - Interactive services
  - Economic advantages
- However, these advantages can be compromised by potentially serious negative side effects
  - Displays may not be as readable as paper manuals
  - Each display may contain substantially less information than a sheet of paper
  - The user interface of online help systems may be novel and confusing to novices
  - The extra mental effort required for navigating through many screen may interfere with concentration and learning, and annotation can be difficult
  - Splitting the display between work and help or tutorial windows reduces the space for work displays
  - Small devices such as cell phones do not have enough display space to provide online help
- Numerous studies have found 15% to 30% slower task times for comprehension or proofreading of text on computer displays, compared to on paper
- Potential Disadvantages in Reading from Displays:
  - Poor fonts, especially on low resolution displays
  - Low contrast between characters and the background
  - Fuzzy character boundaries
  - Emitted light from displays may be more difficult to read by than reflected light from paper
  - Glare may be greater on displays
  - Screen flicker can be a problem
  - Curved display surface may be problem
  - Small displays require more frequent page turning
  - Reading distance can be greater than for paper
  - Displays are fixed in place
  - Display placement may be too high for comfortable reading
  - Layout and formatting problems
  - Reduced hand and body motions with displays as compared to paper may be fatiguing
  - Rigid posture for displays may also be fatiguing
  - Unfamiliarity of displays and the anxiety that the image may disappear can increase stress

9b. OAI Models for Design Manual

Object–action interface, also abbreviated as OAI, is an extension to the graphical user interface, especially related to direct manipulation user interface and it can help to create better human–computer interfaces and increase the usability of a product.

The OAI model graphically represents the users' workplace using metaphors and let the users perform action(s) on the object. The sequence of work is to first select the object graphically (using mouse or other pointing device), and then performing an action on the selected object. The result/effect of the action is then shown graphically to the user. This way, the user is relieved from memory limitation, and syntactical complexity of the actions. Moreover, it emulates WYSIWYG. This feature of OAI lets the user control their sequence of action and visualize the effects at the runtime. If an action results in an undesired effect, the user simply reverses his sequence of actions.



In the action–object model, the computer is seen as a tool to perform different action. Whereas in the object–action model, the user gains a great sense of control from the feeling of a direct involvement. The computer in this case is seen as a medium through which different tools are represented, which is isomorphic to interacting with objects in the real world.

Designing an OAI model starts with examining and understanding the tasks to be performed by the system. The domain of tasks include the universe of objects within which the user works to accomplish a certain goal as well as the domain of all possible actions performed by the user. Once these tasks objects and actions are agreed upon, the designer starts by creating an isomorphic representation of the corresponding interface objects and actions.

The figure above shows how the designer maps the objects of the user's world to metaphors and actions to plans. The interface actions are usually performed by pointing device or keyboard and hence have to be visual to the user so that the latter can decompose his plan into steps of actions such as pointing, clicking, dragging, etc.

This way DMUIs provide a snapshot of the real world situations and map the natural way of user's work sequence through the interface. This means that the users do not have to memorize the course of actions and it reduces the time required to familiarize themselves with the new model of work. Moreover, it reduces the memory load of the users significantly and therefore enhances the usability.

10 a Technical User Manual

A successful user manual provides quick answers in easy to understand terms. The most effective user manuals are organized to give users fast access to information.
The best writers understand their audience. When authoring manuals, writers must think like the end user and understand how the product will be used. This involves anticipating the end user's skill level, most basic needs from the documentation and what problems led to referencing the manual.
Manual content should be written so that it presents the problem, offers a solution and presents methods toward achieving the solution. To achieve this, the writer should focus on the goal of outlining the steps necessary for accomplishing the task at hand.
Here are five important components of effective user manuals:
1. Procedural steps
This is the main body of the user manual, and most likely involve a problem that can't easily be summarized or explained. Logical, numbered steps help the user solve a problem. To aid in clarity, illustrations or videos can help with component or experience visualization.
2. Content clarity
Whenever possible, avoid technical language that may isolate audience segments. Imagine you are writing for a junior high school student. Tables and graphs can help show how sections of content relate, compliment and contrast.
3. Glossary of terms
Every industry has its own language and vocabulary, but some users may not be familiar with even the most commonly used terms. A glossary of terms not only serves as a reference tool, but also gives each term a "home," eliminating the need for the writer to repeatedly re-state and re-explain definitions.
4. Table of Contents
As a general rule, if a manual has more than twelve pages, a table of contents should be used. Entries are listed in order of presentation with accompanying page numbers.
5. Precautionary information
Warnings, notes of caution and danger notices aid user safety and help the manufacturer address liability concerns related to the product. Use universal graphic symbols to represent each type of risk to the end user.
User manual content can range from less than a dozen to hundreds of pages. Generally, the more complex the product, the lengthier the manual. At its core, a user manual is a set of instructions presented in a style and format that facilitates quick reference, and helps the product to succeed in the market.

10 b. Online Tutorial with Animation

Whether in-person or online, facilitating tutorials is an opportunity to work closely with students and understand where they are in their learning. For many graduate students, teaching tutorials is often their first and sometimes only chance to apply and develop their teaching skills.
Tutorials will run differently depending on your discipline; the most common tutorial types are:
Discussion-based tutorials: these tutorials focus on a deeper exploration of course content through discussions and debates.

Problem-solving tutorials: these tutorials are common in math, science and engineering and focus on problem solving processes and quantitative reasoning.
Review and Q&A tutorials: in these tutorials, students ask questions about the course content and assignments, review key course content in preparation for tests or exams, and consolidate their learning in the guiding presence of their instructor or Teaching Assistant (TA).
Numerous aspects are involved in online tutorials and making them productive learning events: planning, communicating, delivery, question strategies, activities, and motivation.

Considerations for Online Tutorials

- Select online tools for running tutorials and inform students about the tools you'll be using. If you are a TA, make this selection in consultation with the course instructor. Provide clear guidelines to students on how online tutorials will work.
- If tutorials are offered in a synchronous format, coordinate with other instructors through your department scheduling rep to avoid scheduling conflicts.
- For synchronous tutorials, provide an outline of the session and explain how students will participate in it. Make sure you know how to share your slides, mute and unmute your microphone, use the chat tool, mute your students' microphones, and share your screen with students.
- For discussion-based tutorials, consider making discussion questions available in advance in LEARN so that students can access the questions if screen sharing does not work.
- If sharing slides in advance via LEARN, share as PDFs, so that students can access the material on their phones.
- For asynchronous tutorials, provide students with the problems or discussion prompts with clear instructions for participation expectations.