Взаимодействие в человеко-компьютерных системах

Лекции 1-2
Введение в дисциплину.
Структура знаний в сфере ЧКВ.
Основные методы анализа и проектирования пользовательских интерфейсов.

Максим Александрович Бакаев,
АВТФ, НГТУ

График работы по дисциплине

Номер недели	Лекционные занятия	Лабораторные занятия
1		
2	**19.02** Лекции 1-2	**19.02** Получение заданий на ЛР2, ЛР3, ЛР4. Получение задания на РГР, обсуждение выбора датасета.
3		
4		**05.03** Самостоятельное выполнение ЛР2.
5		**12.03** Срок сдачи черновика по ЛР2 (задания 1-6)
6		**19.03** Самостоятельное выполнение ЛР2. Самостоятельное выполнение ЛР3.
7		**26.03** Итоговый срок сдачи ЛР2. Срок сдачи черновика по ЛР3 (задания 1-4).
8		**02.04** Самостоятельное выполнение ЛР3.
9	**09.04** Лекции 3-4	**09.04** Итоговый срок сдачи ЛР3. Срок сдачи **черновика РГР** (гипотезы и описание датасета).
10		**16.04** Самостоятельное выполнение ЛР4.
11		**23.04** Срок сдачи черновика по ЛР4 (задания 1-7).
12	**30.04** Лекции 5-6	**30.04** Самостоятельное выполнение ЛР4.
13		**07.05** Итоговый срок сдачи ЛР4.
14	**14.05** Лекция 7	**14.05** Итоговый срок сдачи РГР.
15		
16	**28.05** Защита ЛР2-ЛР4 в форме презентаций.	
17-18	**01-11.06** Прием задолженностей, защита РГР (если требуется).	

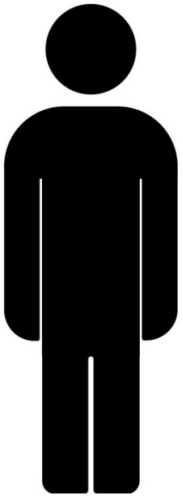
БРС по дисциплине

- Посещаемость:
  - Лекции: 8 * 1 балл
- Лабораторные работы:
  - Выполнение: 3*10 = 30 баллов
  - Защита (ЛР2-ЛР4): 6/12 баллов
- АПМ: РГР, 16 баллов
- АОМ, АТМ: Курсовая работа, 30 баллов
- Экзамен (АПМ): 40 баллов /
  зачет (АОМ): 20 баллов
  - Возможен «автомат» при отсутствии пропусков и соблюдении сроков сдачи отчетов

План лекции

- Введение в ЧКВ
  - Показатели и аспекты взаимодействия
- Структура знаний в сфере ЧКВ
  - Законы, принципы, рекомендации (эвристики)
- Подходы и методы в сфере ЧКВ
  - Процесс разработки ПО и методы анализа и проектирования И.
  - Анализ задач и прецеденты использования (Use cases)
  - A/B тестирование и юзабилити-тестирование

Человеко-компьютерные системы



- Может адаптироваться к окружающей среде
- Обладает знанием и опытом
  - Способен понять и сделать то, чему его не обучали

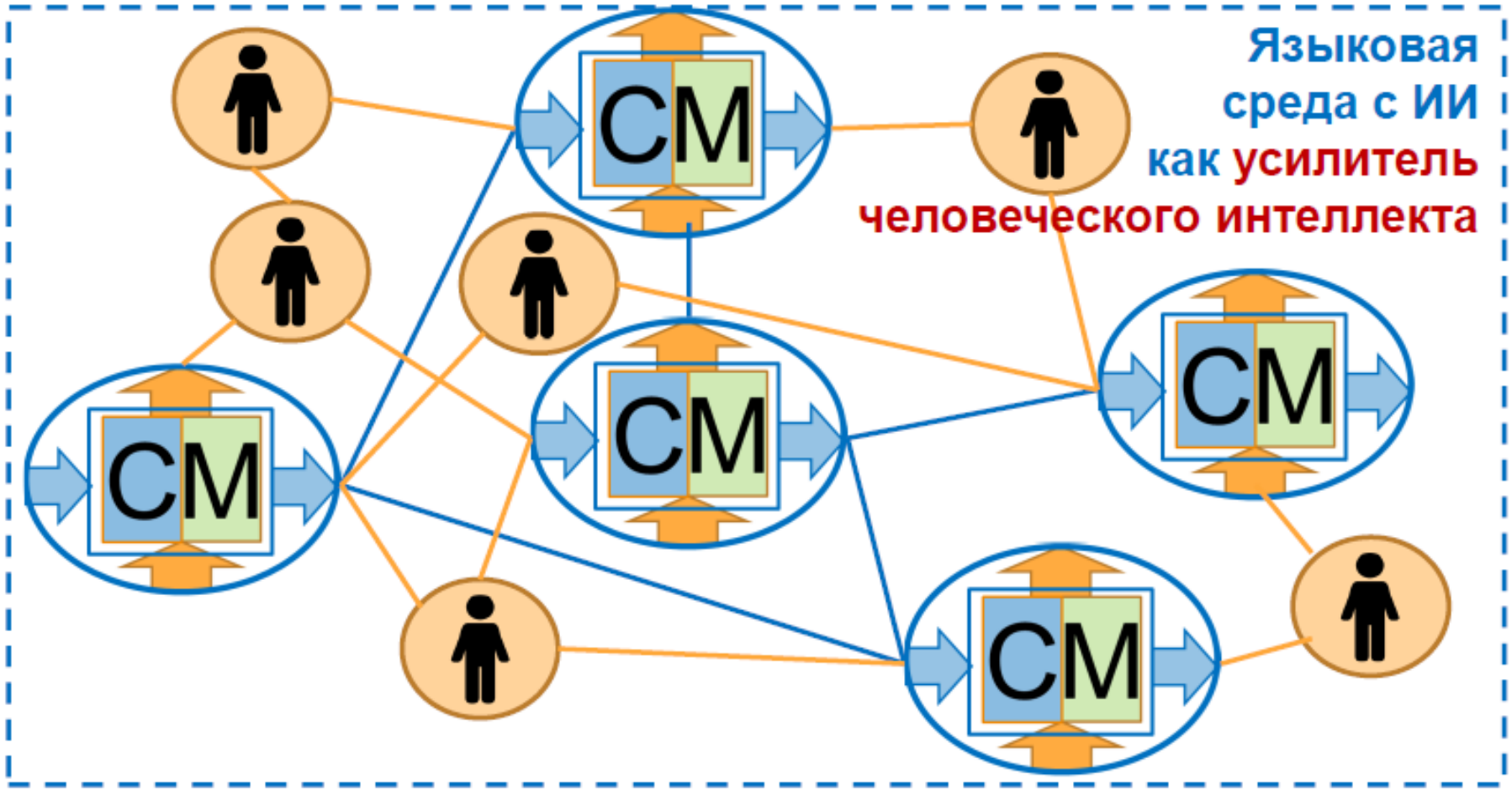
HCI
(ЧКВ)

- Быстро и точно считает
- Может хранить МНОГО данных
- Может работать без остановок

Около 50% всего программного кода в ИС посвящено пользовательским интерфейсам

Human-AI Interaction

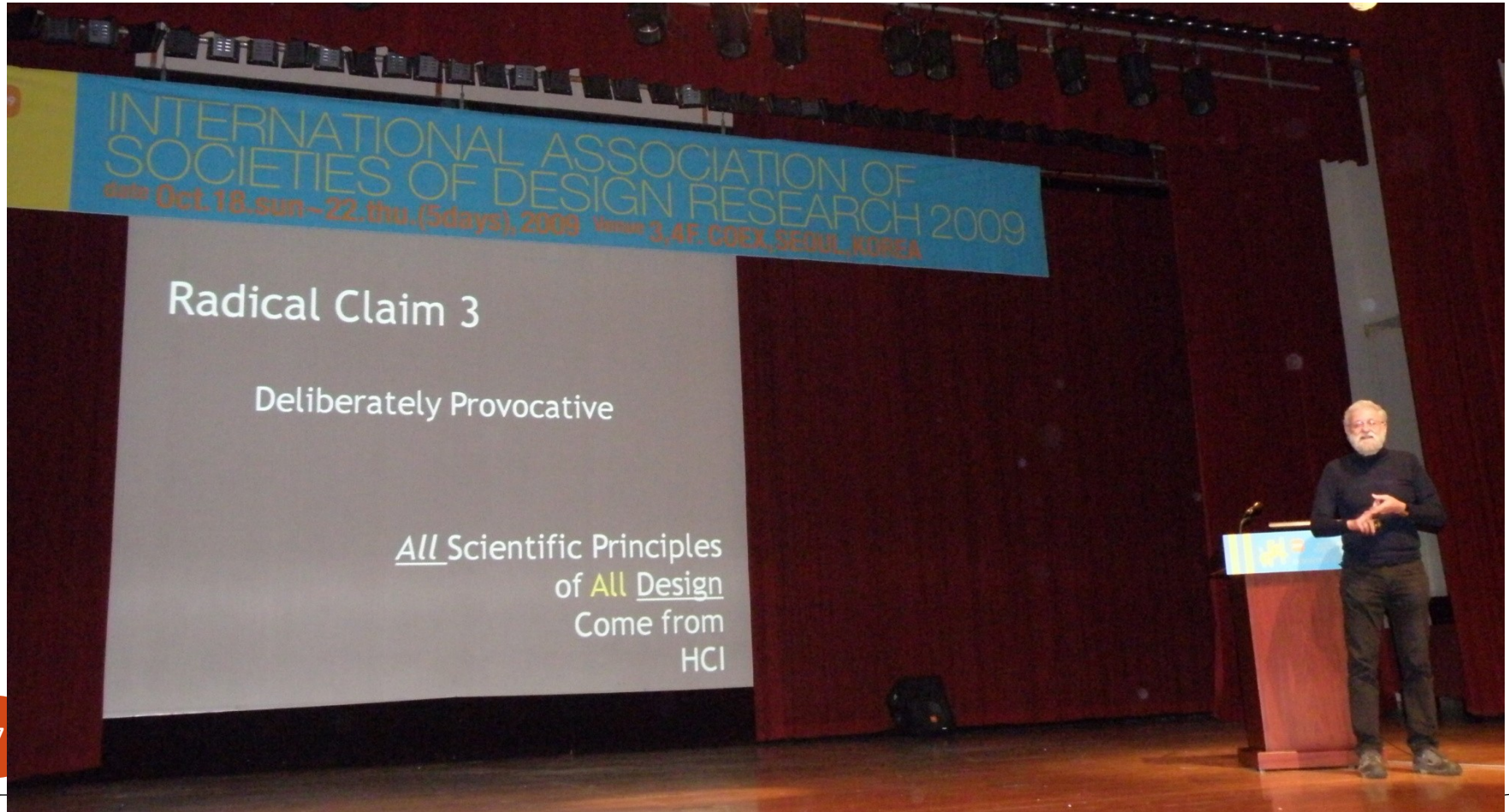
- Ю.В. Визильтер, лекция на Neuroinfo-2023
  - Схема гибридного интеллекта
    - СМ – сократические (рассуждающие) модели



Введение – зачем нужно ЧКВ?

- «Все научные принципы в любом виде дизайна происходят из сферы человеко-компьютерного взаимодействия»

Д. Норман на пленарном докладе IASDR 2009



Введение – история

- Отсутствие выделения интерфейса (взаимодействия) при работе с компьютерами: до 1960-х.
- Развитие и внедрение интерфейсов и устройств:
  - Командная строка
  - Шаровой манипулятор (1949), мышь (1965)
  - Световое перо (1954)
  - Видеодисплеи с векторной графикой (PDP-1, 1961)
  - «Скетчпад» (1963)
  - Графический интерфейс пользователя (Xerox Alto, 1973)
- Проектирование юзабилити – «низкобюджетное» ЧКВ, Я. Нильсен (середина 1990-х).
- «Универсальный дизайн»: с 2000-х .
- Современный рост качества интерфейсов: 6% в год.

Введение – терминология в ЧКВ

- За рубежом:
  - man-machine interaction, computer-human interaction, **human-computer interaction (HCI)**, ~ human factors (in computer systems)
- В СССР (более системный подход, сейчас подзабытый):
  - эргономика (в автоматизированных системах человек-техника, в АСУ), человеко-машинное взаимодействие/общение, инженерная психология
- Дисциплина, изучающая проектирование, оценивание и воплощение интерактивных компьютерных систем, предназначенных для использования человеком, а также сопутствующие аспекты.
- Предмет исследования: **человеко-машинный интерфейс** (ПИ)

Система человек-компьютер

- Человек (оператор):
  - Органы чувств – каналы получения информации
    - Визуальный
    - Слуховой
  - Ввод информации
  - Переработка информации
  - Модель устройства и характеристики памяти
- Компьютер (технические средства АС/АСУ):
  - Интерфейсные устройства
  - Программный слой взаимодействия
- Окружающая среда
  - Факторы внешней среды
  - Контекст взаимодействия
- Показатели качества взаимодействия

Взаимодействие и интерфейс

- Интерфейс – совокупность средств и правил, согласно которым происходит взаимодействие
  - Аппаратные средства: устройства ввода-вывода (интерфейсные устройства)
  - Программные средства: «интерфейсный слой» компьютерных программ (user interface layer) – совокупность элементов, которые могут оказывать влияние на взаимодействие пользователя с программой (её функциями)
  - Правила: язык общения (знаковая система), протокол обмена (коммуникация)
- Условия выполнения оператором функций в АСУ (СССР):
  - наличие информации об управляемом объекте
  - наличие средств отображения этой информации
  - наличие причинной связи между действиями оператора и реакцией объекта, обратной связи
  - наличие цели управления, возможность однозначной реализации управляющего воздействия

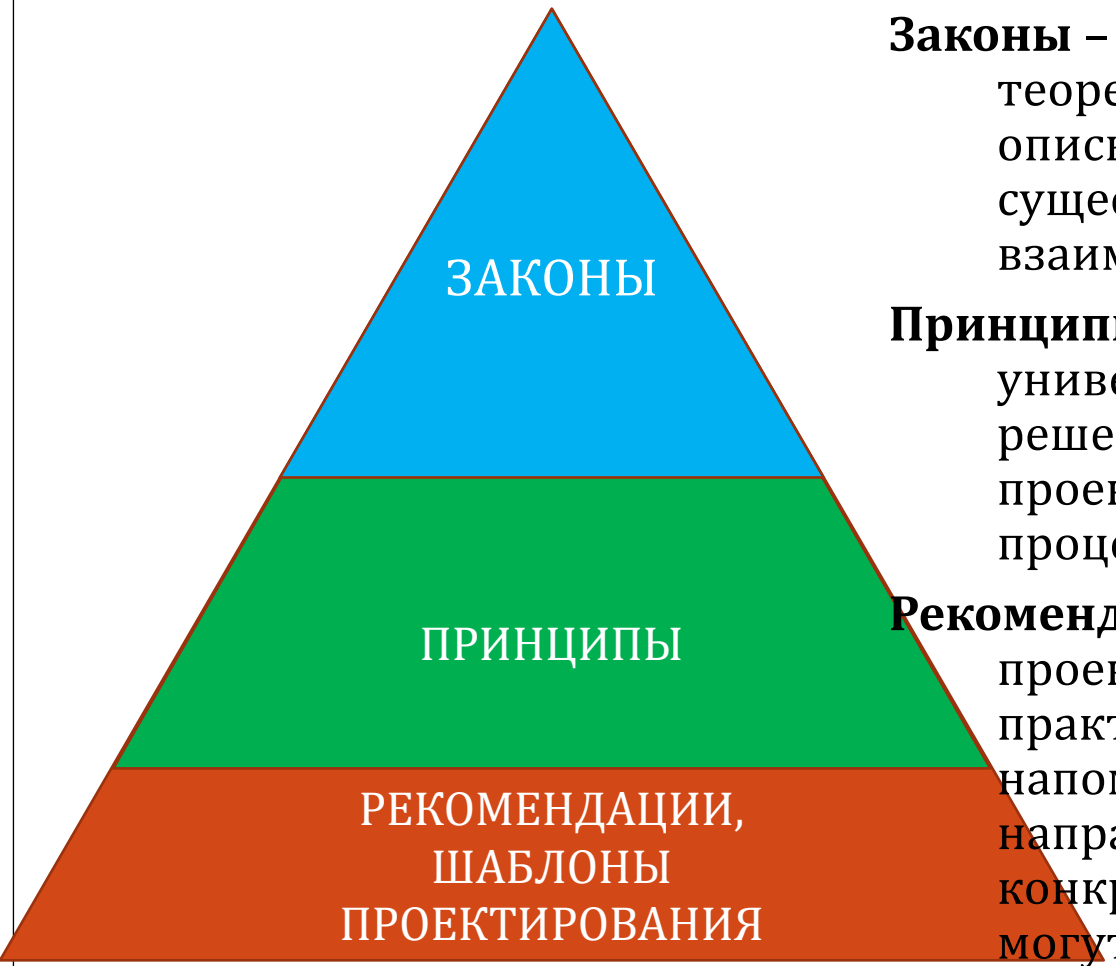
Аспекты взаимодействия

- Физический (с 1940-х годов, в частности в рамках эргономики):
  - Интерфейсные устройства как объекты реального мира
  - Воздействие компьютерной системы на органы чувств человека
  - Пределы выносливости (утомляемость) при различных условиях работы
- Когнитивный (с 1960-х годов: инженерная психология когнитивная психология, лингвистика, бихевиоризм):
  - Восприятие и обработка информации
  - Особенности устройства и работы памяти человека
  - Организация языка общения, понятийной структуры пользователя
  - Обучение, мотивация и т.д.
- Эмоциональный (с 1980-х, ИТ как товар)
  - Субъективные впечатления, возникающие у пользователя (причины возникновения, влияние на работу)
  - «Эмоциональный дизайн», «удовлетворенность пользователя», «проектирование впечатлений пользователя» (User experience design)
- Внимание (с 2000-х, социальные сети и мобильные приложения)

Показатели взаимодействия

- ISO/IEC TR 9126 – Quality in use:
  - эффективность (effectiveness)
  - производительность (productivity)
  - безопасность (safety)
  - удовлетворенность пользователя (satisfaction)
- Якоб Нильсен (качественные характеристики):
  - изучаемость (learnability)
  - эффективность (efficiency)
  - запоминаемость (memorability)
  - ошибки (errors)
  - удовлетворенность (satisfaction)
- Количественные характеристики (стандарт ISO/IEC 25062:2006):
  - процент успешного выполнения задач пользователем (success rate)
  - затраченное на выполнение задач время (performance time)
  - уровень ошибок, допускаемых в процессе выполнения (error rate)
  - субъективная удовлетворенность пользователя (subjective satisfaction)

Существующие знания в сфере ЧКВ



Законы – высокоуровневые теоретические построения, описывающие какие-либо существенные аспекты во взаимодействии или его участников

Принципы (principles) - более или менее универсальные правила, касающиеся решений, принимаемых при проектировании интерфейсов, или процесса проектирования в целом

Рекомендации (guidelines), шаблоны проектирования (design patterns) – практические советы или напоминания, в основном направленные на разрешение конкретных затруднений, которые могут возникать при проектировании, или на предотвращение потенциальных ошибок

Законы в сфере ЧКВ

- Описывают человека (нежели компьютерные технологии), основываясь на его объективных физических и когнитивных возможностях человека (выявлены в рамках когнитивной психологии, бихевиоризма и т.д.)
- Время простой реакции человека:
  - 0.05 с – достаточно, чтобы составить представление о веб-странице
  - < 0.1 с – немедленное действие, эффект «прямого управления»
  - < 1 с – задержка не кажется значительной, но эффект «компьютер отвечает мне»
  - < 10 с – внимание не переключается с задачи (остаётся в кратковременной памяти), но эффект «компьютер тормозит»
  - 30 с – среднее время визита на веб-сайт
  - 1 мин – лимит терпения на большинство несложных задач
  - 2-4 мин – среднее время состоявшегося визита на сайт (если уход, то через 10 с)

Принципы (heuristics) Я. Нильсена

1. **Состояние системы** должно быть всегда понятно для пользователя.
2. Система должна использовать обычный человеческий язык, термины и понятия, существующие в **реальном мире**, а не принятые только внутри самой системы.
3. Пользователь должен **свободно управлять системой**, а не наоборот: пользователи часто ошибаются, поэтому всегда должна быть возможность быстрой отмены нежелательного действия.
4. **Последовательность и единообразие** в системе: пользователь не должен гадать, могут ли разные слова, ситуации или действия означать одно и то же.
5. Тщательный дизайн, **предотвращающий возникновение ошибок**, гораздо лучше, чем выдача сообщений о них. Нужно либо исключить возможность появления ошибок, либо обеспечить проверку их наличия и требовать подтверждения сомнительных действий.

Принципы Я. Нильсена

6. Лучше **понимать, чем вспоминать.** Пользователь должен видеть все объекты и опции, а не держать их в памяти. Инструкции по использованию системы должны быть видны либо легкодоступны.
7. **Гибкость и эффективность использования.** В системе могут быть ускорители взаимодействия, доступные экспертам, но не мешающие неопытным пользователям.
8. **Эстетичный и минималистичный дизайн.** В диалогах не должно быть ненужной или редко используемой информации. Каждый блок дополнительной информации в диалоге конкурирует с действительно необходимой и мешает воспринимать её.
9. Система помогает **распознавать и устранять ошибки.** Сообщения об ошибках должны быть изложены ясным языком, точно указывать на проблему и предлагать конструктивное решение.
10. **Справка и документация.** Лучше всего, если системой можно пользоваться, не читая документацию, но при необходимости нужно обеспечить простой контекстно связанный поиск по справке.

Принципы дизайна для ошибок

- Принципы Д. Нормана:
  - учет возможностей человека (память, нагрузка, …)
  - последовательность поведения системы (концепт. модель)
  - видимые, обозримые варианты и последствия действий
  - представление информации в виде, поддерживающих все типы действий (Skill, Rule, Knowledge), предоставление пользователю «внешней» памяти (технологии, схемы, …)
  - отображение статуса системы, немедленная обратная связь
  - использование ограничений, «ведение» пользователя
  - но предоставление возможностей для экспериментирования
  - признание неизбежности ошибок, разработка механизмов их исправления (отменить действие – просто, совершить что-то необратимое – сложно)
  - стандартизация (в крайнем случае, т.к. удобство – важнее)

Рекомендации (шаблоны, эвристики)

- Знание наиболее конкретного и практического характера
  - Во многих случаях следуют из законов, принципов или эмпирических исследований
  - Учитывают стоящую перед проектировщиком задачу (например, привлечение внимания пользователя к сообщению об ошибке системы).
  - Учитывают контекст дизайна (например, назначение и тип разрабатываемой программной системы, индивидуальные характеристики пользователей и т.д.)
- Проблемы:
  - Значительное количество при слабой организации (затраты на поиск и применение одной – 15 мин) – необходима классификация (категории, теги и т.п.)
  - Соотнесение с контекстом (сложности с интерпретацией в 30% случаев)
  - Полнота и непротиворечивость
  - Конкретность и объективность (желательны ссылки на источник)

Рекомендации и «эвристики» (примеры)

- «Эвристики» – перечень некоторых рекомендаций (реже – шаблонов или принципов), которым должен соответствовать качественный интерфейс или веб-сайт.
- Плохие «эвристики» (примеры):
  - «Сайт должен быть удобен для пользователя»
  - «Дизайн сайта воспринимается позитивно»
  - «Каждый объект имеет понятное пользователю назначение»
- Хорошие «эвристики» (примеры)
  - «Оформление ссылок позволяет понять, что это ссылка»
  - «Название пунктов навигации адекватно отражает содержание соответствующих веб-страниц»
  - «В системе однотипные функции и элементы выглядят одинаково»
  - «Есть доступная функция по работе с обращениями клиентов»

Проектирование Ю. в ИТ-компаниях

- Стадии зрелости компании (по Я. Нильсену):
  1. Враждебность к юзабилити. «Лучший юзер – мертвый юзер!». Ну, или как минимум безрукий.
  2. Юзабилити от разработчиков. Проектировщики полагаются на собственную интуицию при разработке интерфейсов.
  3. Отрывочное проектирование юзабилити. Важность проектирования Ю. осознаётся, но процесса и бюджета нет.
  4. Выделенный юзабилити-бюджет. В основном для юзабилити-тестирования, которое применяется ближе к концу процесса разработки («магический дезодорант»).
  5. Управляемое Ю. Появление юзабилити-менеджера для применения и пропаганды Ю., архива юзабилити-отчётов и т.д.
  6. Систематический Ю.-процесс. Тестирование на ранних этапах, наличие централизованных стандартов и рекомендаций для проектирования, отслеживание показателей качества взаимодействия, связь с бизнесом.
  7. Интегрированный дизайн, направленный на пользователя.
  8. Компания, направленная на пользователя.

Методы и подходы в сфере ЧКВ

- ЧКВ/HCI (исследовательская составляющая):
  - количественный эксперимент (социально-психологические науки) со статистической обработкой данных (дисперсионный анализ, регрессионный анализ, факторный анализ)
  - качественные выводы / рекомендации
- Исследование и проектирование юзабилити (практическая, инженерная составляющая):
  - Анализ задач
  - Прототипирование и итерационный дизайн
  - Экспертное оценивание («эвристический анализ»)
  - Тестирование с реальными пользователям
- Методы ПЮ должны быть **встроены** в процесс разработки ПО: на стадиях анализа требований, проектирования, тестирования, внедрения.

Классические стадии проектирования И.

- В современных методиках разработки процесс итеративный

Популярные методы в ЧКВ

- С привлечением пользователя:
  - Наблюдение за пользователем (Observation) – смотреть, как пользователь работает в своей привычной обстановке.
  - Юзабилити-тестирование – запланированный эксперимент для пользователя (качественный уровень).
  - Непрямые методы:
    - Опросы (Surveys/questionnaires) и интервью (Interviews)
    - Фокус-группы (Focus groups)
  - Протоколирование (для веб-аналитики):
    - Самостоятельное или автоматическое (Яндекс.Метрика, .Вебвизор)
  - Количественные методы (соц.-псих эксперименты)
- Экспертные методы:
  - Эвристическое обследование Ю. – 3-5 экспертов проверяют интерфейс на соответствия общепринятым принципам проектирования («эвристикам»).
  - Анализ задач, «персонажи», сценарии использования, …

Процесс разработки ПО и методы

1. **Начало проекта и анализ требований**
   - анализ контекста использования
   - анализ конкурентов (аналогов, существующих решений)
   - анализ задач пользователя
   - сценарии (прецеденты) использования
2. **Проектирование**
   - прототипирование и итеративный дизайн
   - рекомендации и шаблоны проектирования
   - эвристический анализ
3. **Реализация**
   - таблицы стилей
4. **Тестирование и оценка**
   - юзабилити-тестирование
5. **После релиза (внедрения)**
   - «удалённые» тестирование и оценка

Анализ задач

- Понять, как пользователи достигают **своих** целей
  - Каковы цели пользователей, чего они хотят достичь
  - Что они делают, чтобы этого достичь (процесс работы)
  - Как на пользователей влияют обстановка, знания, опыт
- Технология анализа:
  - Смотреть глазами пользователя!
  - Выделить цель, затем 4-8 задач, каждая из которых имеет подцель (иногда: цель – задача – действия/операции)
  - Определить последовательность выполнения задач, взаимодействие с системой и коллегами, проблемы
  - Задокументировать (диаграммы), сверить корректность
- Проще всего – наблюдение за пользователями
  - Возможно, если есть доступ к пользователям
  - Прямое или удалённое (например, через видео)
  - Главная проблема – не внести искажения присутствием

«Персонажи» («персоны»)

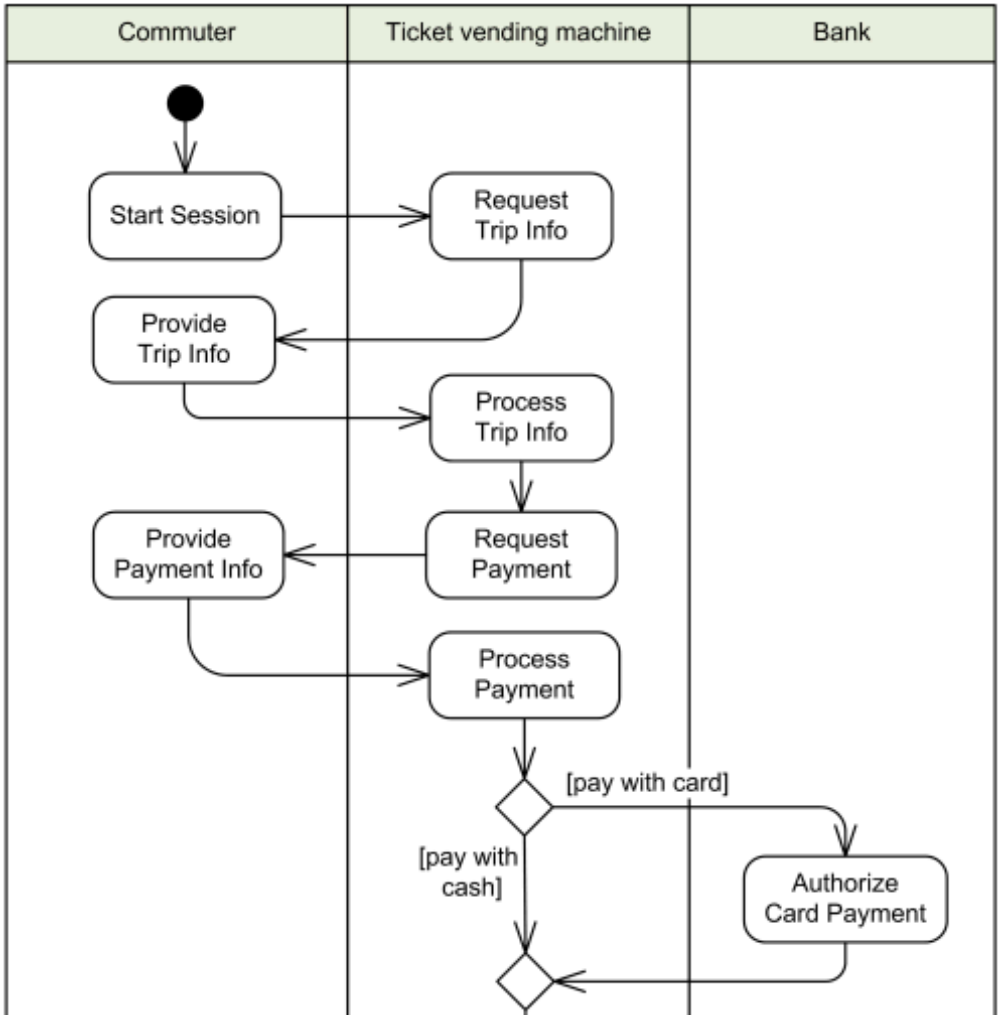
- Помогают:
  - определить функционал и особенности ПО
  - обсуждению и выбору вариантов проекта
  - корректно разработать польз. интерфейс
- Качественно разработанные персонажи (personas)
  - Описывают реальных людей: с биографией, окружением, ценностями. Должны быть запоминающимися.
  - Основываются на качественном анализе (исследование пользователей), иногда и количественном (веб-статистика).
  - Представляют целевую группу пользователей (обычно не более 3-4), отражают её основные цели, чего она ожидает от ПО и как, вероятнее всего, будет его использовать.
- Элементы персонажа: название/целевая группа, имя (можно и фото), род деятельности и проф. обязанности, демография (возраст, пол, образование), цели и задачи в ПО, окружение
  - персонаж – это объект, а не класс!

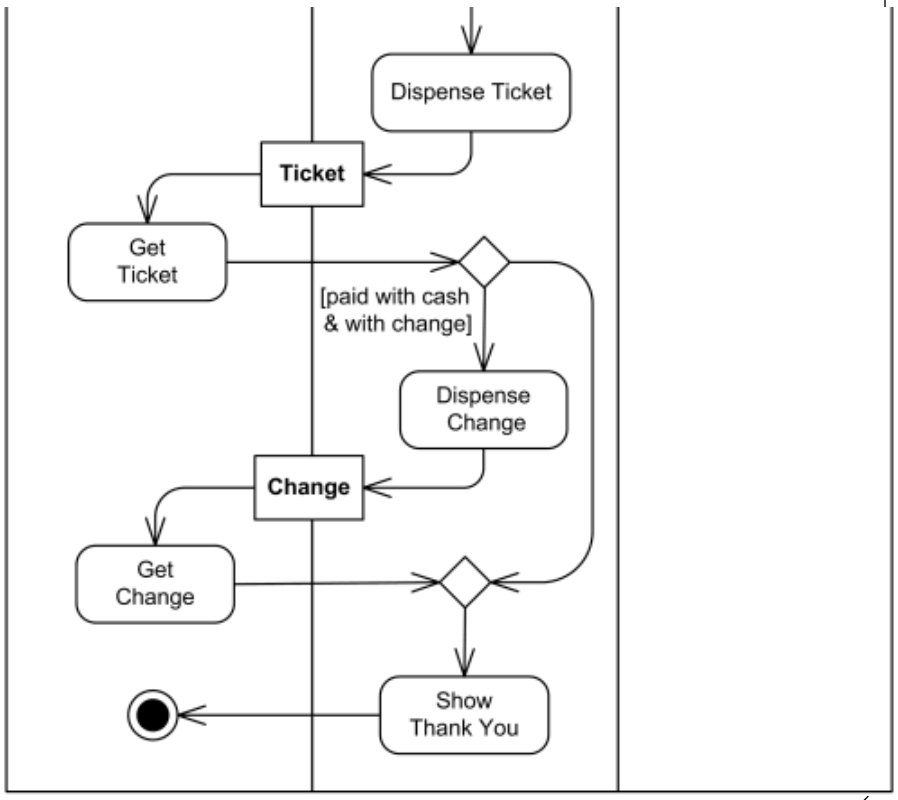
Сценарии (прецеденты) использования

- Сценарии (прецеденты) использования – Use cases:
  - Относятся к функциональным требованиям, соответствуют нижним (рабочим) уровням иерархии задач пользователя
  - Описание поведения системы, её реакции на внешние запросы – кто («актор») и что с ней может делать
  - не затрагивает реализацию (не «как»! – «чёрный ящик»)
  - не описывает пользовательские интерфейсы
  - могут записываться в специализированных нотациях (UML) или обычным текстом, на языке пользователя!
    - номер и название, цель (задача)
    - роли (акторы, персоны), заинтересованные лица (stakeholders)
    - предусловия, активаторы
    - основной сценарий и результат
    - альтернативные пути и исключения

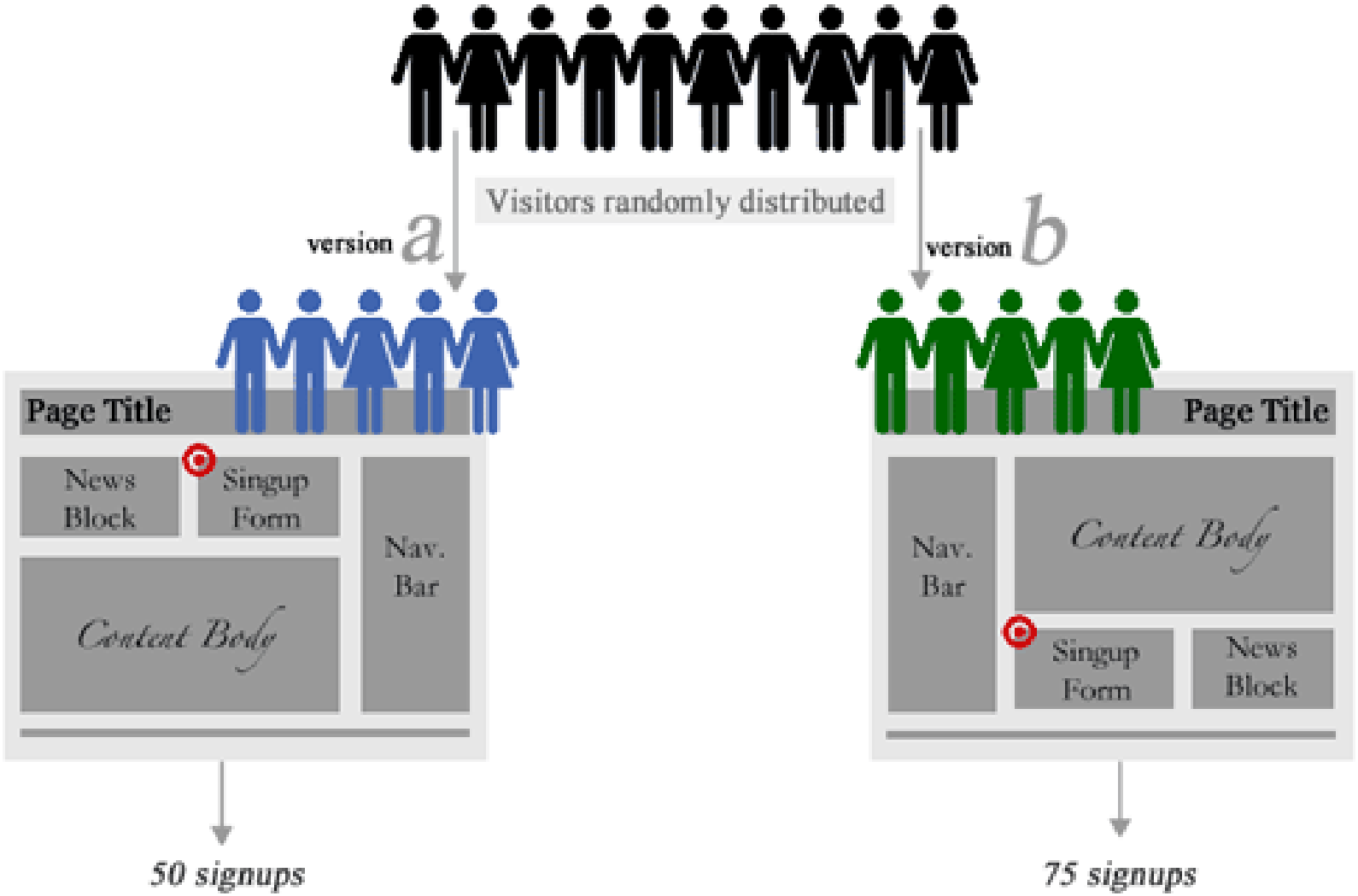
Пример сценария использования

- Покупка билета в автомате





A/B тестирование



Version B is better than version A

A/B тестирование

- Фактически – эксперимент со случайным распределением участников по группам (вариантам дизайна – A и B)
  - Применимы методы статистики (чтобы определить значимость различий между вариантами)
- Обычно отличия вариантов не очень значимы (иначе затруднительно определить повлиявший фактор и тенденции)
- Необходимы готовые (завершённые) дизайны
- Необходимы достаточность и однородность аудитории
- Необходима измеримость KPI (желательно, автоматическая)
- Улучшения KPI обычно не превышают 10%
- Метод применим для относительно мелких элементов

	A/B Testing	Usability	Radical Innovation
Cost	Low	Low–medium	High (unless lucky)
Benefits	1–10%	10–100%	100–1,000%
Risk	None	Low	High
Who can do?	Everybody	Everybody	Geniuses
How often?	Weekly	Monthly	Every 10 years
GDP impact	Medium	High	Medium

Юзабилити-тестирование (ЮТ)

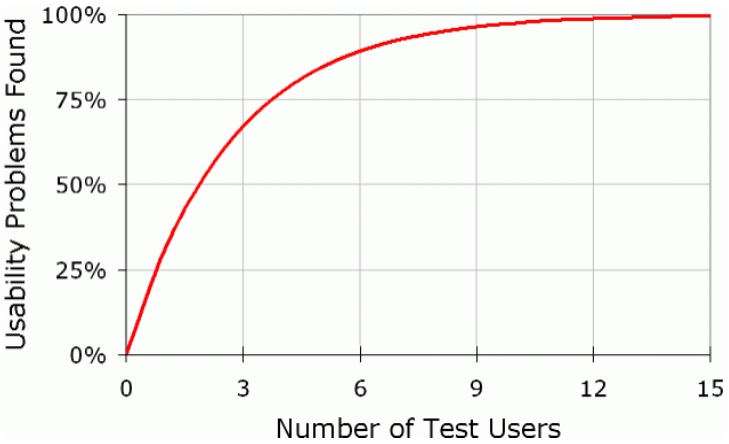
- Цели ЮТ:
  - выявление сильных и слабых мест в интерфейсе для дальнейшего улучшения его в ходе итерационного процесса разработки. Типичный способ – техника «думание вслух».
  - оценка общего качества интерфейса – например, для выбора одного из двух возможных вариантов. Типичный способ – измерение юзабилити.
- ЮТ не является формальным математическим экспериментом; оно не является даже социологическим исследованием.
- ЮТ – метод исследования юзабилити продукта, основанный на привлечении пользователей в качестве тестирующих. Участники, максимально приближенные к реальным пользователям, выполняют типичные для пользователей задачи путем взаимодействия с интерфейсом системы.
- Методы удешевления (с 90-х годов):
  - упрощение понятия юзабилити
  - отказ от сбора количественных данных
  - снижение стоимости оборудования

ЮТ – планирование

- Цель
- Время и место
- Продолжительность
- Необходимое оборудование и софт
- Дополнительные настройки оборудования и софта (начальные экраны, время реакции)
- Ответственный за проведение тестирования
- Количество и состав участников и где их взять
- Количество и состав заданий
- Возможность использования вспомогательных материалов для выполнения заданий
- Признаки успешного и ошибочного выполнения заданий
- Степень допустимого участия (подсказок) со стороны проводящего тестирование
- Набор количественных данных и способы их сбора

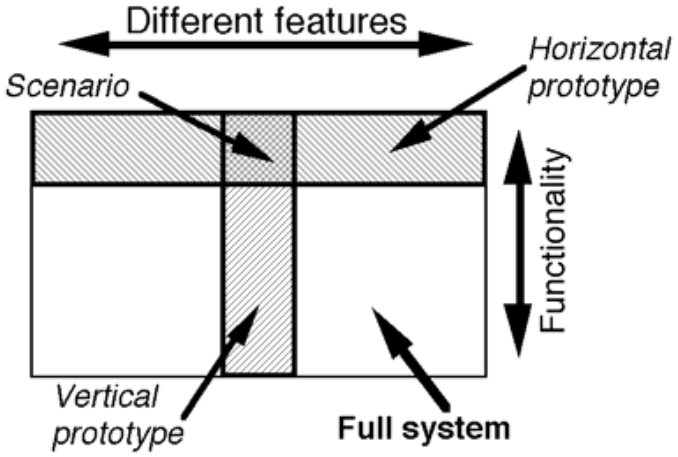
ЮТ – участники

- Количество участников:
  - оптимальное: 3-5
  - если больше, разбить на итерации – проверить внесенные изменения, предлагать более сложные задания.



- Качество участников:
  - максимальное соответствие портрету целевых пользователей продукта (пол, возраст, опыт, социо-экономическое положение и т.д.)
  - эмоциональная открытость (думание вслух)
  - большой опыт в ЮТ – нежелателен

ЮТ – задания



- Качество заданий:
  - максимально приближены к реальным действиям пользователя с продуктом
  - охват наиболее важных частей интерфейса
  - достаточно небольшие (общее время ЮТ 40-90 мин), но не тривиальные, возможны «сценарии»
  - продуманные и опробованные (особо учесть ещё не готовые аспекты), возможны пилотные сессии
- Откуда брать задания?
  - из анализа задач
  - из наблюдения за пользователем
- «Думание вслух» участником
- «Молчание про себя» инструктором

ЮТ – проведение

- Оборудование
  - Компьютер, веб-камера, микрофон (особенно для «думания вслух»
  - Программа записи содержимого экрана (TechSmith Camtasia или Morae), секундомер (если нет в программе)
- Стадии тестирования:
  - Подготовка
  - Введение
  - Непосредственно тестирование
  - «Разбор полетов»
- Пример содержания отчета о тестировании:
  - Резюме
  - Основные проблемы
  - Частные проблемы
  - Количественные данные (если они собирались)
  - Приложение 1. Методика эксперимента и условия теста
  - Приложение 2. Описание тестовых сценариев
  - Приложение 3. Описание участников

Подробнее о ЮТ

- Самая ответственная часть ЮТ (проще всего испортить) – это разработка заданий
- Важная часть – простимулировать участника «думать вслух»
  - Чтобы тот понял и не стеснялся начать, Я. Нильсен рекомендует показать обучающее видео (1 минута – достаточно)
  - Не совсем обычная ситуация. Раскрываются не все типы людей.
  - Необходимо, чтобы пользователь «раскрывал душу», говоря потоком, а не обдумывал, как лучше и красивее сформулировать
  - «Эхо» – повтор фраз пользователя. «Бумеранг» – «ну а как вы сами думаете?». Коломбо» – «то есть вы спрашиваете… эээ…»
- Написание отчёта по ЮТ
  - Результаты ЮТ должны быть юзабельны (для разработчиков)
  - Описывать конкретно (не просто «пользователь затруднился», «пользователь не смог»)
  - Пытаться охватить не только детали, но и общую картину (например, процесс взаимодействия с сайтом в целом)
  - Расставлять приоритеты (критичности) проблем

Дистанционное ЮТ (remote UT)

- Существуют сервисы для проведения удалённого ЮТ (хотя «личное» ЮТ, как правило, предпочтительнее – но могут быть проблемы со временем, бюджетом, удалённостью): Loop11, OpenHallway, TryMyUI, Userlytics, Usertesting.com, www.fabuza.ru, …
- ДЮТ: модерируемое (инструктор «смотрит» за участником в реальном времени) и немодерируемое
- Проблема набора релевантных участников
  - Некоторые современные инструменты ДЮТ предоставляют базу потенциальных участников и возможность выборки из неё
  - Но человек слишком много раз проходивший ЮТ перестаёт быть репрезентативным пользователем
  - Как правило B2B сайты должны находить участников сами
  - «Запас» участников – некоторые могут отказаться или «выпасть»
- Проблема более высоких требований к заданиям
  - Инструктору сложнее скорректировать неправильное понимание пользователем задания