Платформы для оперативной разработки виртуальных лабораторных практикумов для инженерного образования

Тихонов Антон Иванович,
tikhonovai@mpei.ru
Очков Валерий Федорович,
ochkovvf@mpei.ru
НИУ «МЭИ», Москва

Предпосылки

- В январе 2021 года закончилась поддержка технологии Adobe Flash
- Стало затруднительно применять в учебном процессе виртуальные лабораторные практикумы (**ВЛП**) на основе связки Asp.net + Flash
- С середины 2020 начали искать альтернативные решения…

Постановка задачи (1)

- **Найти** связку технологий:
  - Дающих возможность разработки и применения ВЛП в учебном процессе
  - Силами преподавателей и студентов, а не **профессиональных** разработчиков
  - Открытую и бесплатную
  - Простую, не требующую больших усилий для освоения

Постановка задачи (2)

- Требования:
  - Нулевая установка
  - Возможность работы не только стационарных компьютерах и ноутбуках
  - Низкие требования к вычислительной мощности на стороне пользователя
- Этим требованиям удовлетворяют:
  - Веб-приложения
    - На сервере управление пользователями, контентом, заданиями, вычисления
    - Взаимодействие с пользователем на клиенте (пользовательский интерфейс)
- Метафора: волшебная книжка с картинками Г.-Х. Андерсена

Применение в учебном процессе

- Интерактивные электронные учебники
- ВЛП
- Демонстрации на лекциях и практических занятиях (нужно делать быстро и с низкой трудоемкостью)
- Интерактивные приложения для решения задач проектирования (обратных задач)

В чем проблема?

- Разрабатывать ВЛП можно на чем-угодно, но:
  - Высокая трудоемкость
  - Профессиональные разработчики
- Нужны простые технологии возможно за счет выразительных возможностей

Подходы (1): Серверы приложений Matlab, Mathcad, Mathematica

- За:
  - Низкая трудоемкость
  - Простота
  - Для Mathematica имеется книга с десятками примеров ВЛР
- Против:
  - Проприетарные
  - Проблемы с пиковыми нагрузками
  - Очень дорогие

Подходы (2): National Instruments LabVIEW

- За:
  - Низкая трудоемкость
  - Широкий набор виртуальных приборов и устройств
- Против:
  - Проприетарная система, проблемы с лицензиями

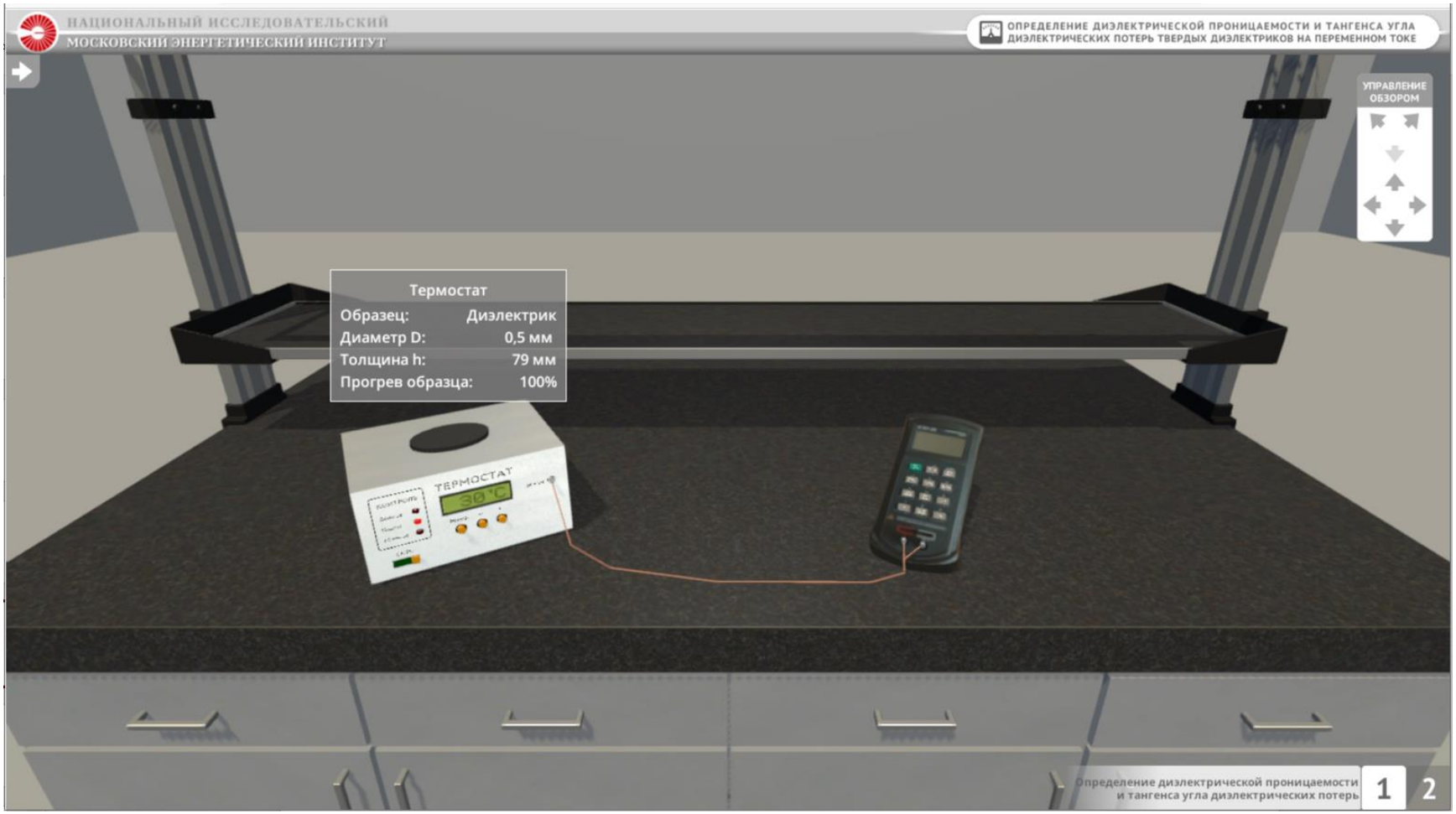
Подходы (3): Asp.Net Core + Blazor

- За:
  - Высокая производительность
- Против:
  - Необходимость создания компонентов ВЛП с нуля
  - Требования к квалификации
  - Проблемы с библиотеками для научно-технических расчетов
  - Зависимость от Microsoft, хотя технология открытая

Подходы (4): Unity

- За:
  - Все необходимое для создания игрового контента, в том числе и 3D
- Против:
  - Требования к квалификации
  - Высокая трудоемкость
  - Проприетарный продукт

Подходы (4): Unity



Подходы (5): Java + frontend

- За:
  - Высокая производительность
  - Открытость
- Против:
  - Необходимость создания компонентов ВЛП с нуля
  - Требования к квалификации
  - Высокая трудоемкость

Подходы (6): R + Shiny

- За:
  - Простота
  - Низкая трудоемкость
  - Открытость
- Против:
  - Ограниченность (задачи статистики)

- Подход очень привлекательный, была сделана попытка переноса Shiny в экосистему Python…

Подходы (7): Экосистема Python

- За:
  - Простота
  - Низкая трудоемкость
  - Открытость и бесплатность
  - Широчайший набор библиотек
  - Можно привлекать студентов…
- Против:
  - Сравнительно низкая производительность
  - Отсутствие «серебряной пули» в качестве технологии для разработки и создания ВЛП

Jupyter (1)

- За:
  - Интегрированный вычислительный документ (markdown+latex+графика+анимация+ вычислительный эксперимент) – все «в одном флаконе»
  - Простота (пользовательский интерфейс – две строки)
  - Открытость, бесплатность
  - Замечательно для проведения занятий в дистанционном формате
  - Возможность работы на тонком клиенте (JH, TLJH)

Jupyter (2)

- Против:
  - Необходимость поддержки постоянного (keep alive) соединения (мобильные провайдеры)
  - Медленный старт
  - Трудности с созданием интегрированных приложений

Аналоги Jupyter (приложения Python в браузере)

- Panel
- Streamlit
- PyScript
- Mercury

Что предлагается

- Серверная часть **Django** (наличие наработок и готовых приложений)
- Виртуальные лабораторные работы: **Dash** (Ajax «из коробки»)
  - Компоненты (функционал+GUI) на Python
  - Повторно используемые компоненты на React
- Взаимодействие Django-Dash – django-plotly-dash через WSGI, ASGI, Redis

Особенности Dash

- Выполнение на клиенте и сервере без перезагрузки страниц
- Функции обратного вызова (callbacks)
- Большой набор компонентов
- Возможность достаточно простого создания пользовательских компонентов
- Сборочный подход:
  - Раскладка пользовательского интерфейса (компоненты) на основе Grid Layout
  - Остается написать функции обратного вызова для обеспечения функциональности

Построение ВЛП

- Учебный контент (тексты, формулы, видео) Django
- Управление пользователям – Django (интеграции с университетской системой пока нет)
- Управление контентом, заданиями – через подгружаемые электронные таблицы Excel (с последующим сохранением в СУБД)
- ВЛР Dash (встраиваются в iframe и/или непосредственно в веб-страницу)
- Разработка в основном под Windows
- Собранная система Virtual Box VM под Ubuntu Server

Технология разработки ВЛР (участие студентов (*))

- Реализация математической модели (Python) (*)
- Подготовка данных (Excel, Python) (*)
- Компоненты Python для сокращения числа callbacks и повторного использования (*)
- Компоненты React (один раз участвовал студент) для виртуальных приборов и устройств
- Сборка ВЛР из компонентов (*)
- Тестирование (*)
- Методическое обеспечение
- Интеграция в ВЛП

Предложенная связка технологий позволяет:

- С умеренной трудоемкостью разрабатывать ВЛР и интерактивные веб-приложения силами непрофессиональных команд
- Интегрировать ВЛР в электронные учебники
- Привлекать к разработке студентов

Спасибо за внимание!