Informatik Spektrum (2022) 45:106–114
https://doi.org/10.1007/s00287-022-01447-2

HAUPTBEITRAG

Erstellung einer kollaborativen Arbeitsumgebung für die pandemiebedingte Online-Lehre — ein Praxisbericht

Tobias Tefke[1] · Elias Petri[1] · Ralf C. Staudemeyer[1] · Timo Kucza[1]

Angenommen: 2. Februar 2022 / Online publiziert: 5. April 2022

Zusammenfassung
Durch die pandemische Lage sind viele Lehrangebote nicht in ihrer ursprünglichen Form durchführbar. Im hier vorgestellten Fall soll das Programmieren einer Hardware-Entwicklungsplattform für das Internet der Dinge (IoT) erlernt werden. Die Studierenden arbeiten in Arbeitsgruppen mit bis zu fünf Teilnehmern. In der dezentralen Lehre ist dies nur schwer umsetzbar. Im vorliegenden Fall hätten dann drei Geräte pro Teilnehmer ausgegeben werden müssen. Ein Austausch innerhalb der Projektgruppen war zunächst nur eingeschränkt per Videokonferenz möglich. Das fragliche IoT-Labor wurde deshalb in eine virtuelle Umgebung integriert, so dass dort in Gruppen programmiert werden kann, während die IoT-Hardware mittels Video in Echtzeit beobachtet wird. Die geschaffene Lehrumgebung vereint die Durchführung von Vorlesungen, Übungen, Tutorien und Arbeitsgruppentreffen. Die Möglichkeiten zur Interaktion erleichtern es den Studierenden unter anderem, während der Vorlesung in Kontakt miteinander zu treten.

Kollaborative Gruppenarbeit mit Labor-Hardware ist mit vorhandenen Online-Konferenzlösungen kaum datenschutzkonform zu realisieren. Um den Zugriff auf vorhandene Geräte für Arbeitsgruppen zu ermöglichen, wurden bestehende FLOSS[1]-Lösungen zusammengeführt und erweitert. Die Funktionalität sowie erste Erfahrungen zur Entwicklung und Nutzung werden hier vorgestellt.

Einleitung

Die Corona-Pandemie stellt die praxisorientierte Lehre vor besondere Herausforderungen. Jederzeit muss mit Kontaktbeschränkungen ohne langfristige Ankündigung gerechnet werden. Dementsprechend müssen Lehrveranstaltungen auf dezentrale Durchführung umstellbar sein. Durch die räumlichen Begrenzungen der PC-Pools und Labore sind diese von den Regelungen besonders hart getroffen. Eine Nutzung der Räumlichkeiten ist im ursprünglichen Sinne teils gar nicht mehr möglich. Insbesondere problematisch sind Lehrveranstaltungen, in denen die Studierenden in Arbeitsgruppen direkt mit spezieller Hardware interagieren müssen. Dabei muss der Zugang zur Hardware des Labors ermöglicht und die soziale Interaktion innerhalb der Arbeitsgruppen unterstützt werden.

In der hier vorgestellten Veranstaltung sollen Studierende das Programmieren einer hardware-basierten Internet of Things (IoT)-Entwicklungsplattform erlernen und praktisch üben. Die Plattform besteht dabei aus einer Linux-basierten Entwicklungsumgebung und mehreren Industrie-zertifizierten IoT-Geräten und Sensoren, die im Rahmen eines eigenen Projektes miteinander interagieren sollen. Hierbei handelt es sich um Kleinstgeräte mit einer mit Mikrocontrollern vergleichbaren Rechenleistung. Die Projektgruppen haben einen Umfang von je fünf Studierenden. Vor der Pandemie wurde im Labor an den Geräten gearbeitet, was regelmäßig auch einen intensiven fachlichen Austausch zwischen den Studierenden zur Folge hatte.

Die Umsetzung dieser Lehrveranstaltung in der Online-Lehre war schwierig. Um individuelle Übungen von zu Hause zu ermöglichen, müssten jedem Studierenden drei IoT-Geräte inklusive Sensoren zur Verfügung gestellt werden. Allerdings ist hierfür der Hardware-Bestand der Fakultät nicht ausreichend und eine entsprechende Beschaffung war finanziell nicht möglich. Den Studierenden ist die Ein-

[1] https://www.gnu.org/philosophy/floss-and-foss.en.html (Zugegriffen: 07.02.2022).

✉ Tobias Tefke
t.tefke@stud.fh-sm.de

Ralf C. Staudemeyer
r.staudemeyer@hs-sm.de

[1] Fakultät Informatik, Hochschule Schmalkalden, Am Schwimmbad, 98574 Schmalkalden, Deutschland

richtung einer eigenen Entwicklungsumgebung aus Kostengründen ebenfalls häufig nicht möglich.

Selbst wenn die IoT-Geräte in ausreichender Stückzahl vorhanden wären, wäre eine Kollaboration kaum möglich, da dann jeder Studierende individuell im eigenen Tempo arbeitet und somit ein Austausch über Lehrinhalte nur begrenzt über Videokonferenzen möglich wäre. Ein Problem bei der Verwendung bestehender kommerzieller Kollaborationssoftware ist häufig, dass diese hinsichtlich des Datenschutzes bedenklich ist. Gleichzeitig ist eine Datenschutzgrundverordnung (DSGVO)-konforme Umsetzung zwingend notwendig. Eine zentrale Anforderung an unser Kollaborationssystem besteht entsprechend darin, allen Nutzern einen bestmöglichen Datenschutz zu gewähren. Es sollen generell nur die Daten erhoben werden, welche für den Betrieb zwingend notwendig sind.

Eine Lösung der beschriebenen Probleme war mit den verfügbaren Anwendungen für die digitale Lehre nur begrenzt möglich. Daher haben wir uns entschieden, mit Hilfe bestehender quelloffener Lehrwerkzeuge eine eigene, zusammenhängende und einfach bedienbare Plattform zu entwickeln. Diese adressiert die vorgestellten Probleme und ist im hohen Maße anpass- und erweiterbar. Damit ist eine Adaption an andere Lehrformate möglich. Der Einsatz von Free/Libre and Open Source Software (FLOSS) garantiert diese Erweiterbarkeit, die beispielsweise im Rahmen von Projektarbeiten mit Studierenden umgesetzt werden kann.

Durch FLOSS lassen sich weiter zentrale Fragen des Datenschutzes und der Datensicherheit adressieren. Nur FLOSS ermöglicht den Zugriff auf die Programmquellen und damit die Durchführung von aussagekräftigen Sicherheitsprüfungen. Wie schon bei eigenen Weiterentwicklungen eröffnet dies Möglichkeiten sowohl zur externen Beauftragung als auch zur internen Durchführung mit Studierenden.

Im Sinne der Datenvermeidung ist es naheliegend, eigene Server für lokale Dienste zu betreiben. Nur auf diesem Wege kann vermieden werden, dass die Metadaten der Verbindungen auch außerhalb des Einflussbereiches der Hochschule anfallen. Alle von uns verwendeten Tools nutzen FLOSS-Lizenzen, welche die genannten Bedingungen erfüllen (Apache 2.0, GNU LGPL v3.0, AGPL).

Wir stellen im Folgenden zunächst dar, wie wir unser real existierendes IoT-Labor virtualisiert haben. Dies betrifft das Bereitstellen der kompletten Linux-Entwicklungsumgebung nebst Videostreams der IoT-Hardware für sechs Gruppenarbeitsplätze mit bis zu jeweils fünf Teilnehmern. Im Anschluss wird das Programm WorkAdventure (WA) eingeführt, welches wir als FLOSS-Plattform verwenden, um die Anwendungen für Vorlesung und Laborarbeit zugänglich zu machen. Abschließend wird gesondert auf die relevanten Aspekte von IT-Sicherheit und Datenschutz des virtuellen Labors eingegangen, bevor wir die Ergebnisse unserer Arbeit zusammenfassen.

Technische Infrastruktur

Dieser Abschnitt beginnt mit einem Überblick über die Rahmenbedingungen an unserer Fakultät. Dann folgen erste Details zu den notwendigen Eigenentwicklungen. Zunächst wird die Virtualisierung von Einzel- und Gruppenarbeitsplätzen für einen Linux-PC-Pool vorgestellt. Dann beschreiben wir die Entwicklung der Labor-Arbeitsplätze mit angeschlossener IoT-Hardware und deren Livestream. Im Anschluss gehen wir auf diverse von uns überwundene Herausforderungen ein.

Rahmenbedingungen

Bereits vor Anfang der Pandemie war Jitsi Meet [4] vielen Mitgliedern unserer Fakultät für Online-Meetings bekannt (Apache 2.0-Lizenz). Der Funktionsumfang umfasst die spontane Erstellung von Konferenzräumen und ermöglicht neben Audio- und Videoübertragung auch das Teilen von Bildschirminhalten. Verwendet wurde Jitsi zunächst vornehmlich für spontane Absprachen, da durch zahlreiche öffentliche Jitsi-Meet-Server die Nutzung lediglich einen Browser erforderte. Die Erstellung eines virtuellen Konferenzraumes ist für alle Teilnehmer ohne Verwaltungsaufwand möglich.

Für Vorlesungen wurde zum Sommersemester 2020 das Tool Big Blue Button BigBlueButton (BBB) [2] als hochschulweiter Dienst eingeführt (GNU LGPL v3.0-Lizenz). BBB bietet im Vergleich zu Jitsi Meet weitgehende Funktionalitäten für die Lehre.

Genutzt wird insbesondere eine Präsentationsfläche, in der Vortragsfolien hinterlegt werden können. Diese hat Schreib- und Zeichenwerkzeuge für die gemeinsame Bearbeitung, sowie Funktionalitäten zum Erstellen von geteilten Notizen. Weitere oft genutzte Funktionen sind das Beantragen von Redezeit durch „Handheben“ sowie die Möglichkeit für spontane Teilnehmerumfragen.

Hervorzuheben ist, dass die Vortragsfolien vom Lehrenden in das BBB-System hochgeladen, dort in das virtuelle Whiteboard eingebunden und dann im Browser des Nutzers lokal gerendert werden.

Das Gespann aus BBB für die Vorlesungen, Übungen und Tutorien sowie Jitsi Meet für sonstige Absprachen und virtuelle Treffen ermöglicht es uns bereits, die meisten Veranstaltungen angemessen als Fernlehre durchzuführen. Lehrveranstaltungen, welche vor der Pandemie in unseren PC-Pools und Laboren durchgeführt wurden, stellen damit jedoch weiterhin ein Problem dar.

Durch die Abstandsregelungen sind Gruppenarbeiten im Labor nicht mehr durchführbar. Die Aufgabe bestand also darin, entferntes Arbeiten und Kommunizieren möglichst realitätsnah so wie im Labor zu ermöglichen. Neben der bereits durch die genannten Programme ermöglichten Kommunikation musste weiter eine Möglichkeit geschaffen werden, den Arbeitsgruppen einen gemeinsamen Zugriff auf die Geräte zu ermöglichen.

Die Hardware im Labor mit einer Kamera zu erfassen und so den Gruppen zugänglich zu machen, war hierzu ein naheliegender erster Schritt. Dies bietet gemeinsamen visuellen Zugriff auf die begrenzt verfügbare Hardware. Aber wie die Programmierung als Gruppe durchgeführt werden könnte, blieb damit zunächst weiter offen. Notwendig ist die gemeinsame Benutzung eines Laborrechners, der per USB mit der IoT-Hardware-Entwicklerplattform verbunden ist. Auf diesen Rechnern ist eine spezielle Entwicklungsumgebung mit Werkzeugen für die spezifische Hardware installiert.

Für die Programmierumgebung suchten wir ein programmunabhängiges FLOSS-Tool für den grafischen Fernzugriff, dessen Client im Browser des Nutzers lauffähig ist. Grundsätzlich ist dies mit einem VNC-Server und dem in JavaScript geschriebenen Client noVNC realisierbar. Bei der Nutzung von Virtual Network Computing (VNC) werden der Bildschirminhalt sowie die Maus- und Tastaturereignisse zwischen dem lokalen und dem entfernten Rechner ausgetauscht. Somit war eine Lösung für das grafische Arbeiten an einem entfernten Laborrechner gefunden.

Virtualisierung der PC-Arbeitsplätze

Um PC-Arbeitsplätze zu virtualisieren, bestehen grundsätzlich zwei Optionen: Entweder die Ein- und Ausgaben der Rechner aus den tatsächlichen PC-Pools direkt weiterleiten oder in einem sogenannten Hypervisor einen virtuellen PC-Pool einrichten und diesen dann für alle Arbeitsplätze nutzen. Da die Virtualisierung aller Rechner eine erhebliche Vereinfachung bei der Pflege und Wartung versprach, haben wir uns für diesen Weg entschieden. Entsprechend werden die Arbeitsplätze der virtualisierten PC-Pools auf einem zentralen Serversystem betrieben.

Als Hypervisor nutzen wir Quick Emulator (QEMU) [9] gemeinsam mit libvirt [5]. Die virtuellen Maschinen in den PC-Pools verwenden dabei als Festplatte ein gemeinsames Basis-Abbild. Änderungen an diesem sogenannten „base image“ während der Nutzung der Maschinen werden mittels des Dateisystem-Features Copy-on-Write (CoW) in eine separate Datei geschrieben, die dann als „disk image“ bezeichnet wird. CoW wird dabei durch QEMU mit dem Dateiformat qcow2 realisiert. Innerhalb der virtuellen Maschinen wird das Linux-Dateisystem ext4 verwendet.

Das Zurücksetzen der virtuellen Maschinen ist dann einfach. Hierzu muss lediglich die „disk image“-Datei neu erzeugt werden. Dadurch werden alle gespeicherten Nutzerdaten vernichtet.

Durch den Einsatz von CoW wird der verursachte Ressourcenverbrauch erheblich reduziert, da nur geänderte Dateisystemblöcke separat gespeichert werden müssen. Dadurch ist es uns möglich, auf einem HP Proliant DL 380p Gen8 mit nur 128 GB Arbeitsspeicher 27 vollwertige Linux-Arbeitsplätze zeitgleich auf 100 GB Festplattenspeicher anzubieten. Dies erfolgt zusätzlich zur Bereitstellung der kompletten Lernumgebung, auf die wir später noch näher eingehen.

Den Zugriff auf die virtuellen PC-Arbeitsplätze haben wir über das VNC-Protokoll realisiert. Über einen durch QEMU bereitgestellten VNC-Server können Nutzer mit einem VNC-Client auf den Bildschirminhalt des virtuellen Arbeitsplatzes zugreifen und diesen mit Maus und Tastatur aus der Ferne bedienen. Der Zugriff über das VNC-Protokoll ist Grundbestandteil des Hypervisors und erfordert daher zunächst keine weiteren Anpassungen.

Entfernter Zugriff auf die IoT-Hardware

Die sechs Laborrechner, auf welchen die virtualisierte IoT-Hardware-Entwicklungsumgebung läuft, haben im Gegensatz zu den Rechnern in den PC-Pools jeweils einen eigenen Hypervisor.

Um Arbeiten an der im Labor real existierenden IoT-Hardware-Entwicklungsumgebung zu ermöglichen, muss diese für die entsprechenden virtuellen Maschinen erreichbar sein. Im vorliegenden Fall geht es dabei um Zolertia Re-Motes [6]. An jede Instanz sind jeweils drei dieser Geräte per Universal Serial Bus (USB) angeschlossen und vom jeweiligen Hypvervisor mittels USB-Passthrough an die dort laufende virtuelle Maschine (VM) durchgereicht.

Die Re-Mote-Plattform basiert auf einem 32-Bit ARM Cortex-M3 System-on-Chip (SoC) (TI CC2538) mit 32 MHz Taktfrequenz, 512 KB programmierbarem Flash und 32 KB Arbeitsspeicher. Zur Datenübertragung hat die Re-Mote eine IEEE 802.15.4 kompatible integrierte 2,4-GHz- und zusätzlich eine 868/915-MHz-Funkschnittstelle (TI CC1200). Dabei ermöglicht das 868/915-MHz-Frequenzband eine Kommunikationsreichweite von bis zu 2 km.

Auf den mit diesen Re-Motes verbundenen Rechnern (insgesamt sechs IoT-Arbeitsplätze) wurde analog zum Serversystem jeweils eine vollständige Virtualisierungsumgebung installiert. Das Abbild für diese Maschinen umfasst eine fertig eingerichtete Entwicklungsumgebung mit allen Werkzeugen zum Programmieren der Geräte und bietet dabei Zugriff auf die angeschlossene, reale Hardware.

Die virtualisierte Entwicklungsumgebung läuft auf Basis des Linux-Betriebssystems Lubuntu 20.04. Im Kern besteht

sie aus den fertig eingerichteten Programmierwerkzeugen für den ARM Cortex-M3 und die MSP430-Plattform. Als IoT-Betriebssystem wird dabei Contiki-ng [3] verwendet. Ebenfalls installiert ist Cooja, ein Contiki-Emulator, mit dem sich erste Übungen realisieren lassen. Weiter ist das Lernmaterial aus Übungen und Tutorien hinterlegt, welches von den Studierenden als Anleitung genutzt werden kann. Die virtualisierten IoT-Arbeitsplätze sind per VNC nutzbar.

Um die Videoüberwachung der Hardware zu realisieren, wurde an jeden IoT-Arbeitsplatz eine über USB-Passthrough an die VM durchgereichte Videokamera angebunden. Diese ermöglicht die Verfolgung der Light Emitting Diode (LED)-Aktivitäten an den Re-Motes und damit den aus dem realen Labor gewohnten Blick auf die Geräte. Für den Zugriff auf die Re-Mote wird parallel dazu die auf der VM installierte Entwicklungsumgebung genutzt. Der Videostream wird dabei bandbreitenschonend über die VNC-Verbindung übertragen.

Integrationsbedarf

Damit sind die gewünschten Funktionen (Konferenzen zur Kommunikation sowie Zugriff auf Arbeitsplätze und die spezielle Hardware) technisch verfügbar, aber ohne Integration zunächst nur schwer nutzbar. Weiteres Ziel war es deshalb, ein integriertes und dabei einfach bedienbares System zu erstellen, das komfortablen Zugriff auf alle wesentlichen Funktionen bietet. Dieses sollte dabei an einer zentralen Stelle zugänglich sein. Technische Details, wie etwa die Abbildung der Gruppen auf unterschiedliche Konferenzräume, sollten möglichst ohne zusätzlichen manuellen Verwaltungsaufwand gelöst werden. Wir haben WorkAdventure verwendet, um die bereits dargestellten Funktionen für Benutzer zugänglich zu machen. Diese Umsetzung beschreiben wir im Folgenden.

WorkAdventure-Plattform

Die Lösungen für Vorlesungen, PC-Pool und IoT-Arbeitsplätze sind unabhängig voneinander und lassen sich parallel, auch in mehreren unterschiedlichen Veranstaltungen, nutzen. Da allerdings Bedarf an einer einzelnen, integrierten Plattform bestand, mussten die Einzellösungen nun noch zusammengeführt werden. Idealerweise so, dass es möglich ist, fließend zwischen den einzelnen Lösungen zu wechseln. Hierzu setzen wir eine mit Avataren begehbare Karte ein. Die Anwendungen sind in Teile dieser Karte eingebunden, so dass mit dem Betreten des jeweiligen Kartenteils durch einen Benutzer die entsprechende Anwendung in dessen Browser geöffnet wird.

Aufwendige 3D-Welten waren dabei ausgeschlossen, da die erstellte Lösung ohne hohe Anforderungen auch auf älteren Geräten performant laufen muss. Als Basis für die weitere Entwicklung verwendeten wir das FLOSS-Tool WorkAdventure (WA) [10] (AGPL v3-Lizenz), welches diese Funktionalität in Form einer 2D-Welt bietet und vollständig im Browser läuft.

Einführung in WorkAdventure

WA ist ein Browser-Programm, mit dem sich Arbeitsumgebungen erstellen lassen. In einer virtuellen Umgebung bewegen die Teilnehmer individuell konfigurierbare Avatare auf einer 2D-Karte. Das allgemeine Erscheinungsbild von WA ist dem eines 16-Bit-Spieles der frühen 90er Jahre nachempfunden. Eigene Karten können mit dem Tiled Map Editor [11] erstellt werden. Die Karten müssen dann anschließend exportiert und auf einen Webserver kopiert werden. Wir nutzen dabei den in WA integrierten Apache-Webserver. In WA können die Karten anschließend über den URL eingebunden werden.

Sobald sich mehrere Figuren auf einer Karte nah beieinander stellen, wird aus diesen automatisch eine Kommunikationsgruppe gebildet. Hierbei werden Kommunikationskanäle erstellt, in denen die Nutzenden miteinander kommunizieren können. Diese „Kanäle“ werden dabei dezentral aufgebaut, d. h. direkt zwischen den Browsern der Teilnehmer. Dies ermöglicht eine weitgehend private Kommunikation. Als visuelle Indikation der Gruppenzugehörigkeit wird um die Teilnehmer ein Kreis gelegt (wie bei den Studierenden im Vorlesungssaal links oben in Abb. 1 ersichtlich).

In der Kommunikationsgruppe ist es dann möglich, über Audio, Video und einen Textchat miteinander zu kommunizieren, aber auch den Bildschirminhalt zu teilen. Die Gruppen passen sich dabei dynamisch den Aktionen der Benutzer auf der Karte an. Zunächst ist es durch die Avatare überhaupt erst möglich, andere Teilnehmer und Gruppen zu sehen. Und dann ist es jederzeit möglich, durch Weggehen eine Gruppe zu verlassen und sich durch Hinzustellen einer anderen Gruppe anzuschließen. Insbesondere diese Funktionalität ist für die Abbildung der Gruppenarbeit außerordentlich wertvoll. Der unkomplizierte Übergang zwischen diesen Gruppen durch entsprechende Positionierung des Avatars ist dabei natürlicher, als etwa Gruppen in einer Liste nachzuschlagen und diesen dann irgendwie beizutreten.

Kartenteile können in WA mit Aktionen hinterlegt werden, die dann aufgerufen werden, wenn ein Benutzer den entsprechenden Teil der Karte betritt. Eine für WA typische „Aktion“ ist etwa das Öffnen einer Webseite. Hierbei wird die Umgebung von WA visuell geteilt: auf der linken Seite wird im Browser weiterhin WA angezeigt. Auf der rechten Seite wird die neue Webseite eingebunden. Hierzu kommt das Hypertext Markup Language (HTML)-Element

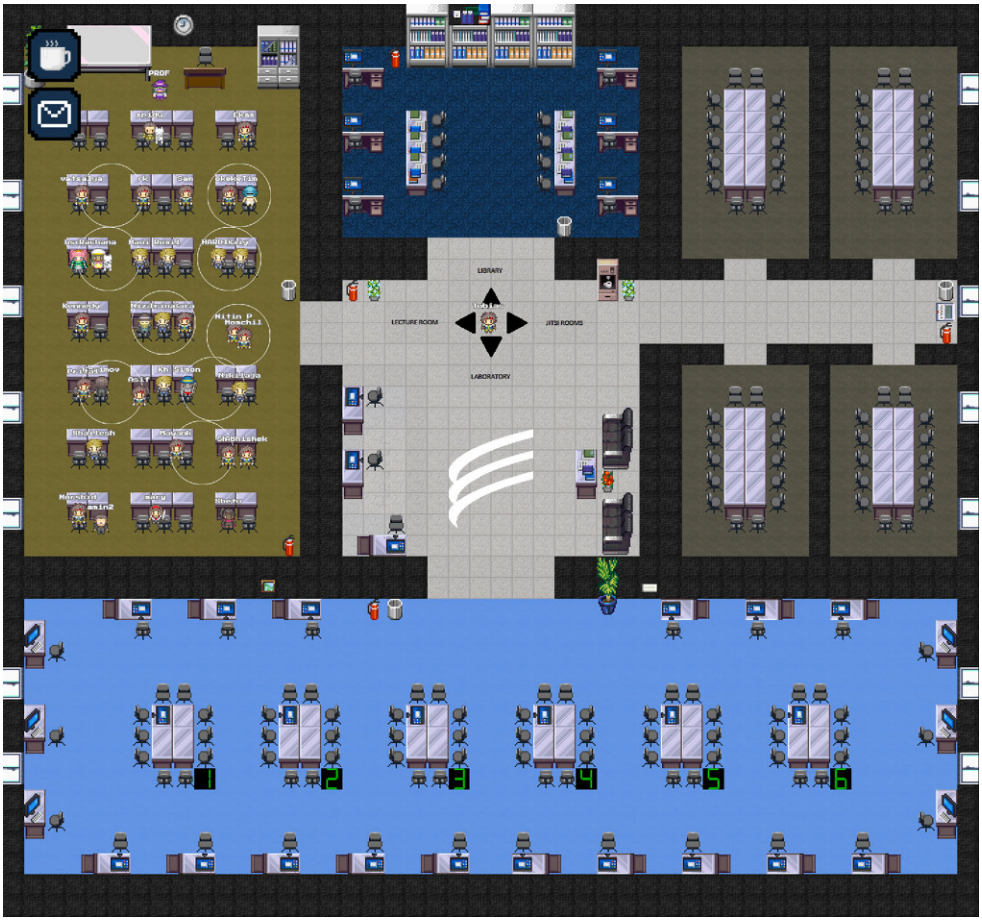

Abb. 1 Eingangskarte mit Lobby (*in der Mitte*), Jitsi-Räumen (*rechts*), Vorlesungssaal (*links*) und IoT-Labor (*unten*)

„iFrame“ zum Einsatz. Damit ist die Einbindung von webbasierten Diensten einfach zu bewerkstelligen. In WA eingebundene Webseiten werden auch als „CoWebsites“ bezeichnet. Über die Scripting-API von WA lässt sich weiter der gesamte Funktionsumfang von JavaScript verwenden. Die Scripting-API ermöglicht Anpassungen im Layout und im Verhalten der aktuellen Karte, ähnlich zu DOM-Manipulationen im Browser.

Erstellte Karten und Funktionalität

Um unser Labor virtuell angemessen abzubilden, haben wir eigene Karten erstellt. Im Folgenden beschreiben wir diese Karten sowie deren Verwendung. Des Weiteren wird darauf eingegangen, wie wir die Infrastruktur in diese integriert haben.

IoT-Labor

In der Mitte der Karte befindet sich zunächst die Lobby (siehe Abb. 1). Beim Betreten wird der Avatar des Teilnehmers automatisch in diesem Bereich platziert. Die Lobby kann dabei für Kommunikationsgruppen von bis zu vier Personen genutzt werden. Von hier aus kann der Benutzer die weiteren Bereiche der Karte betreten. Diese werden in den folgenden Absätzen kurz beschrieben.

Links von der Lobby befindet sich der erste Vorlesungssaal. Für dessen Realisierung wurde BigBlueButton (BBB) [2] in WA eingebunden. Beim Betreten des BBB-Bereiches wird über das Backend eine Anfrage an die BBB-API gesendet. Als Antwort wird eine URL, welche zu der dem Raum zugeordneten Vorlesung führt, zum Webclient des jeweiligen Benutzenden gesendet.

Benutzer müssen lediglich ihren Avatar in den virtuellen Vorlesungssaal bewegen, um in die BBB-Konferenz der Vorlesung zu gelangen. Aus Nutzersicht öffnet sich in WA eine CoWebsite mit BBB. In dem BBB-Raum findet die dem Raum zugeordnete Veranstaltung statt (siehe Abb. 2). Um eine versehentliche Teilnahme zu vermeiden, muss die Aktion manuell durch das Drücken der Leertaste bestätigt werden. Mit dem Verlassen des Vorlesungsraumes wird die Konferenz dann automatisch verlassen.

Über dem Lobby-Bereich befindet sich eine Bibliothek. In diese ist die Katalogsuche der Hochschulbibliothek als CoWebsite integriert. Von hier aus kann die Literatur, welche in der Vorlesung vorgestellt wurde, in Form von E-Books online gelesen und teilweise auch im PDF-Format heruntergeladen werden. Rechts von der Lobby sind vier

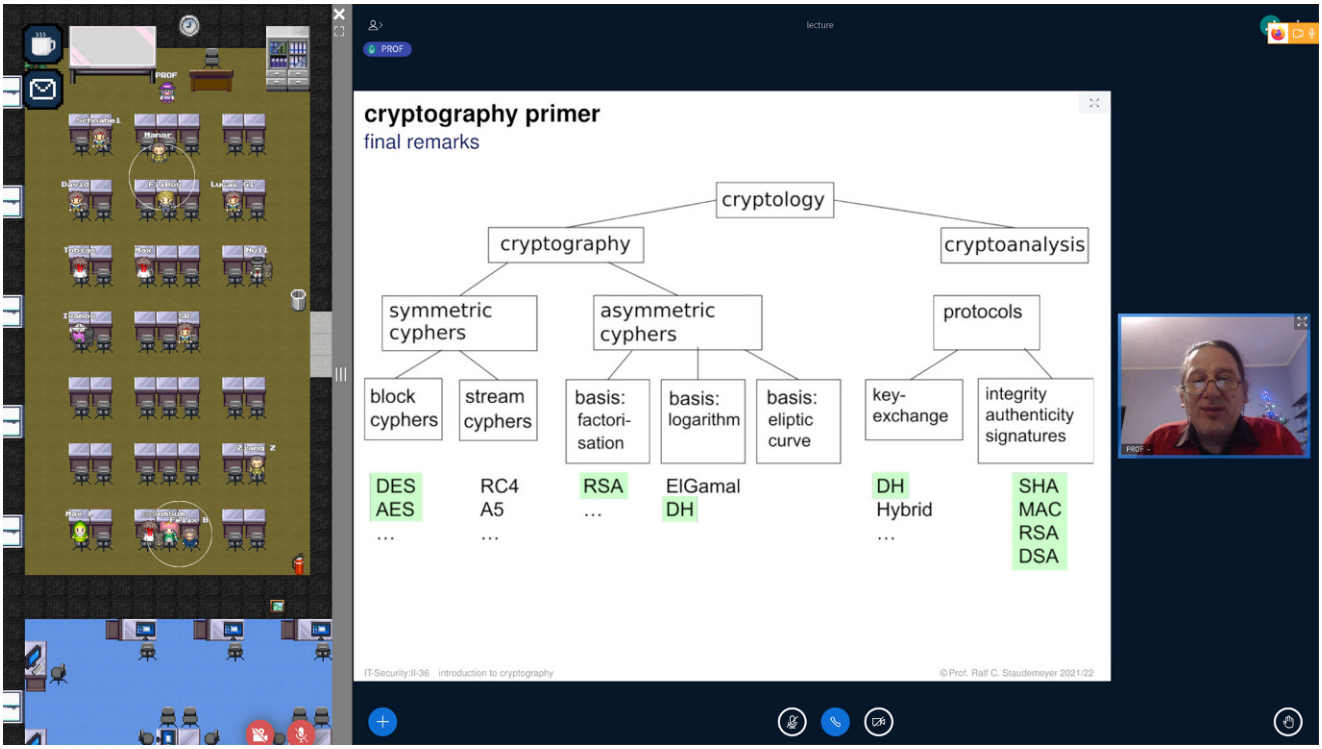

Abb. 2 Laufende Vorlesung via BBB im Vorlesungssaal

Jitsi-Meet-Räume eingerichtet. Diese können bei Bedarf als Seminar- oder Übungsräume genutzt werden.

Der untere Teil der Karte beinhaltet den Bereich des IoT-Labors für Übungen an den Einzel- und Gruppenarbeitsplätzen. In diesem Bereich wurde die Kleingruppen-Kommunikation in WA deaktiviert, um ein versehentliches Stören beim Vorbeilaufen an Arbeitsplätzen zu verhindern. Für Arbeitsgruppen ist die Kommunikation bei Nutzung des Gruppenarbeitsplatzes via eingebettetes Jitsi realisiert. Gruppen können sich bei Bedarf ebenfalls in der Lobby treffen oder einen der Meet-Räume nutzen.

Einzelarbeitsplätze

Um die per VNC erreichbaren PC-Arbeitsplätze einzubinden, wurde noVNC [8] als JavaScript-basierter VNC-Client integriert, der dann im Webbrowser des Benutzers ausgeführt wird. Dabei wird noVNC als CoWebsite eingebunden. Um den Bildschirminhalt darzustellen, greift noVNC über ein serverseitiges Backend (Websockify) auf den VNC-Port der virtuellen Maschine zu. Websockify ermöglicht es noVNC, über einen Websocket mit der virtuellen Maschine zu kommunizieren [8, 12].

Zur Authentifizierung des Benutzers auf dem virtuellen Rechner wird dann ein von WA abgefragtes (Kurs-)Passwort an noVNC übergeben [7]. Hierzu wurde WA um eine Funktion erweitert, mit der das Passwort an noVNC übergeben wird.

Für die IoT-Veranstaltung wurde auf den Einzelarbeitsplätzen der Cooja Simulator [1] installiert. Hierbei handelt es sich um ein Programm, mit welchem emulierte IoT-Geräte programmiert werden können. Diese sollen den Studierenden am Emulator in den ersten Übungen ein eigenständiges Einarbeiten in die IoT-Programmierung ermöglichen. Es stehen den Studierenden aktuell insgesamt 22 Einzelarbeitsplätze für die zeitgleiche Nutzung zur Verfügung.

Gruppenarbeitsplätze mit Hardwarezugriff

Die Gruppenarbeitsplätze ermöglichen die Arbeit an den Geräten im IoT-Labor unserer Fakultät. Es sind insgesamt sechs Gruppenarbeitsplätze vorhanden, welche in Form von Tischen in der Mitte des Raumes abgebildet sind.

Tritt ein Avatar an einen solchen Arbeitsplatz heran, wird eine Verbindung zu der entsprechenden VM hergestellt (siehe Abb. 3). Die Arbeitsfläche in WA wird dabei aufgeteilt. Links ist der Kartenteil sichtbar, in dem sich der Nutzer aktuell befindet. In der Mitte wird der virtuelle Arbeitsplatz eingebunden. Am rechten Rand wird die Jitsi-Konferenz angezeigt.

An Gruppenarbeitsplätzen wird die VM über eine selbst entwickelte Schnittstelle (Multi-User-VM-Assigner) eingebunden, die es ermöglicht, dass mehrere Nutzer gleichzeitig an einer VM arbeiten und miteinander kommunizieren können. Diese Schnittstelle wird ebenfalls als CoWebsite

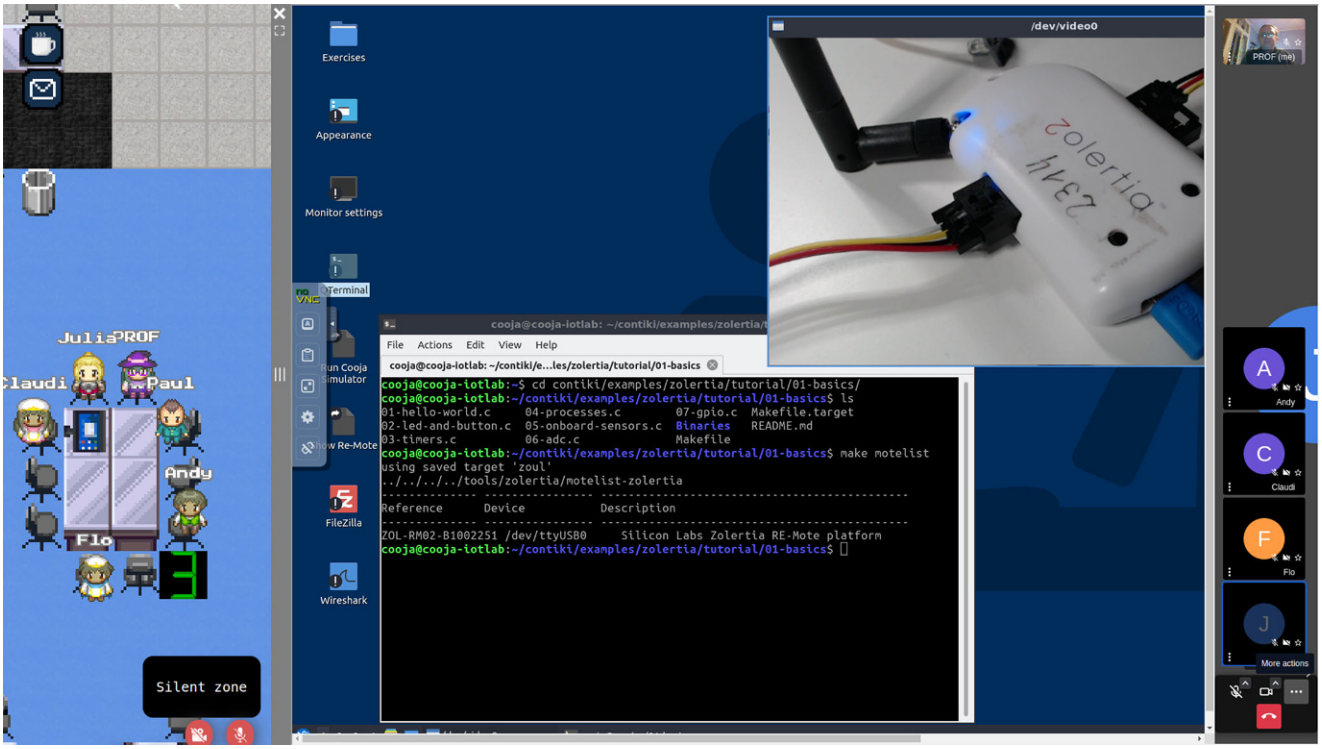

Abb. 3 Gemeinsames Arbeiten an einem IoT-Gruppenarbeitsplatz mit Blick auf die Entwicklungsumgebung und IoT-Hardware

in WA eingebunden. Sie ermöglicht die Teilnahme an einer Jitsi-Konferenz sowie den Zugriff auf den Gruppenrechner mittels noVNC. Indem in der Schnittstelle zwei HTML-iFrames statt nur einem genutzt werden, wird sichergestellt, dass nur die Gruppenteilnehmer mit diesem noVNC-Zugriff im entsprechenden Jitsi-Raum sind.

Administrationsschnittstelle

Zusätzlich zur FLOSS-Version bietet die Entwicklerfirma TheCodingMachine noch Zugriff auf eine proprietäre, kostenpflichtige Administrationsschnittstelle an. Mittels dieser ist es beispielsweise möglich, Benutzerkonten mit bestimmen Rechten und entsprechenden Einladungslinks zu erstellen. Damit kann u. a. der Zugriff auf die Karten beschränkt werden. Allerdings bedeutet der Kauf der Zusatzfunktionen auch, dass Nutzerdaten auf den Servern von WA anfallen. Dies wollten wir jedoch vermeiden.

Deshalb wurde von uns eine Administrationsschnittstelle mit der für uns notwendigen Funktionalität implementiert. Mit dieser Schnittstelle werden Zugriffsbeschränkungen zu Karten realisiert. Es können Benutzer- bzw. Gruppenkonten erstellt und Zugriffsrechte vergeben werden. Dies sind insbesondere Zugangsrechte zu Vorlesungsräumen und Laboren, aber auch Sonderrechte für (Gast-)Dozenten. So ist es mit entsprechender Berechtigung möglich, Nachrichten an bestimmte oder gar alle Nutzer zu versenden oder eigene Texturen für Avatare zu hinterlegen.

In Abb. 4 wird die technische Struktur des virtuellen Labors dargestellt. Die Pfeile geben dabei an, ob die Komponenten uni- oder bidirektional miteinander kommunizieren.

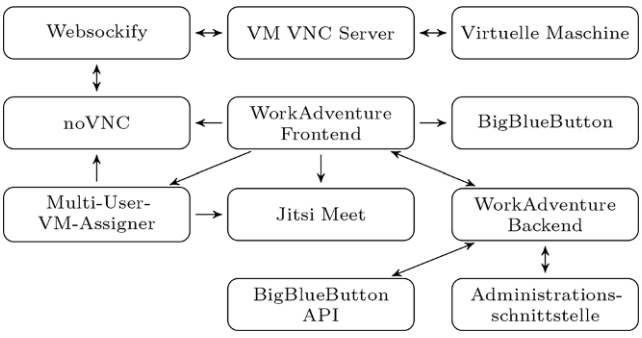

Abb. 4 Struktur des virtuellen Labors

IT-Sicherheit und Datenschutz

Ein wichtiger Aspekt bei der Entwicklung des virtuellen Labors war es, den Datenschutz ernst zu nehmen und die Daten der Nutzer bestmöglich zu schützen. Dem Grundsatz der Datensparsamkeit folgend sollten überhaupt nur die Daten erhoben werden, die für die Nutzung des Labors zwingend notwendig sind. Diese müssen dann bestmöglich geschützt werden. Auf diesen Aspekt wird in diesem Abschnitt zuerst eingegangen. Danach wird die Autorisie-

rung der Benutzer beim System vorgestellt. Im Anschluss werden die Sicherheitsprobleme, welche von Studierenden gefunden wurden, präsentiert.

Datenschutz

Von unserem System werden nur zwingend erforderliche Daten erhoben. Hierbei handelt es sich auf der Serverseite neben einer Benutzer-ID um die IP-Adresse sowie einen frei gewählten Benutzernamen. Die IP-Adressen werden nur temporär im Arbeitsspeicher gespeichert, ausschließlich um die Kommunikation mit den verbundenen Webclients zu ermöglichen. Statt realer Namen können Nutzer ein beliebiges Pseudonym wählen. Der Benutzername und die ID werden auf dem Server ebenfalls nur temporär für die Dauer der Verbindung vorgehalten.

Benutzerdaten, welche zur Konfiguration von WA und dem virtuellen Charakter notwendig sind, werden ausschließlich in den Browsern der Benutzer gespeichert. Die lokal gespeicherten Daten beinhalten u. a. die Benutzer-ID, den Namen, den ausgewählten Charakter und die zuletzt besuchte Karte. Um diese Daten zu entfernen, kann der Nutzer die Daten der Laborwebsite in seinem Browser löschen.

Autorisierung

Um die Nutzer beim System zu autorisieren, werden Einladungslinks ausgegeben. Dies ist erforderlich, um unbefugten Nutzern einen Zugriff auf die Laborumgebung zu verwehren. Mit diesen Einladungslinks wäre es möglich, einen Benutzer eindeutig zuzuordnen. Um eine Nutzerzuordnung durch Anonymisierung zu erschweren, werden Einladungslinks für Nutzergruppen ausgegeben. Dies ist zwar von WA nicht so vorgesehen, allerdings beeinträchtigen die damit verbundenen Seiteneffekte das Nutzererlebnis nur unwesentlich.

Sobald sich ein Nutzer mit einem Einladungslink authentifiziert, wird mittels der von uns implementierten Administrationsschnittstelle geprüft, ob die im Link hinterlegte Benutzer-ID vergeben ist. Ist dies nicht der Fall, wird der Zutritt verwehrt. Andernfalls werden die für den Nutzer eingestellten Rechte von der Administrationsschnittstelle abgefragt und gewährt. Im Anschluss kann der Nutzer die virtuelle Umgebung betreten.

Identifizierte Sicherheitsprobleme

Während der Entwicklung der Lernplattform wurden im Rahmen einer Lehrveranstaltung von Studierenden Sicherheitsprobleme identifiziert. Zwei werden exemplarisch im Folgenden kurz vorgestellt.

Denial of Service

Durch eine fehlerhafte Rechteprüfung war es möglich, das Backend von WA zum Absturz zu bringen. Standardnutzer konnten über die Entwicklerwerkzeuge der Browser eine versteckte Eingabemaske öffnen, die nur für Administratoren sichtbar sein sollte. Das Öffnen war über die Änderung eines Attributs an einem HTML-Element möglich. Beim Verschicken von Befehlen überprüft WA die Berechtigungen. Dabei wurde auch korrekt festgestellt, wenn der Nutzer nicht über die entsprechende Berechtigung verfügt. Fälschlicherweise führt dieser Fall jedoch im Backend zu einer Fehlermeldung, die nicht behandelt wurde und damit zum Absturz führte.

Information Gathering

Bei WA-Instanzen ohne Administrationsschnittstelle bestand die Möglichkeit, Debug-Informationen abzurufen. Dies ermöglichte es beispielsweise herauszufinden, welche und wie viele Nutzer gerade in der jeweiligen WA-Instanz online sind. Das Abrufen der Debug-Informationen erfolgt über eine HTTP-Anfrage. Hierzu muss ein Token zur Authentifizierung übergeben werden. Dabei handelt es sich um dasselbe Token, mit dem sich das WA-Backend bei der Administrationsschnittstelle authentifiziert.

Bei WA-Instanzen, welche keine Administrationsschnittstelle verwenden, wurde das Token auf einen Standardwert gesetzt. Sofern dieser nicht geändert wurde, war es somit möglich, sich mit dem Standardwert zu authentifizieren und die entsprechenden Daten vom Backend auszulesen.

Fazit

Es ist uns gelungen, eine Lehr- und Lernumgebung mit hoher Interaktivität zu erstellen und dabei bestehende Hardware aus unserem IoT-Labor erfolgreich einzubinden. Durch die konsequente Nutzung von FLOSS konnten tiefgreifende Anpassungen am Quelltext vorgenommen und diese nun der Allgemeinheit zur Verfügung gestellt werden. Sowohl der Quelltext des Projektes als auch die von uns neu erstellten Karten sind auf GitHub frei verfügbar[2].

Die von uns entwickelte umfassende Plattform bietet die Möglichkeit zur einfachen Anpassung und Nutzung auch in anderen Lehrveranstaltungen. Die Integration weiterer Werkzeuge analog zu unserem Vorgehen bietet weitgehende Individualisierbarkeit und so das Potential für weitere Erweiterungen. Der Betrieb eines eigenen Servers verhindert dabei, dass Benutzerdaten bei Herstellern bzw. Diensteanbietern anfallen.

[2] https://github.com/SUASecLab.

Im Sinne der Datensparsamkeit haben wir die Plattform dahingehend angepasst, dass sämtliche Benutzerdaten ausschließlich im Browser des Nutzenden gespeichert und somit auch jederzeit durch diesen gelöscht werden können. Serverseitig werden von uns nur noch zwingend notwendige Benutzerdaten, wie Benutzer-ID und IP-Adresse, erhoben und dann auch nur für die Dauer der Verbindung vorgehalten.

Während der Entwicklung des Labors haben Studierende mehrere Sicherheitslücken in den verwendeten Programmen entdeckt. Diese wurden gemeldet und sind inzwischen behoben.

Das im persönlichem Austausch geäußerte, positive Feedback der Studierenden bestätigt uns, dass die Umgebung ein gutes Lernklima ermöglicht. Das Labor unserer Hochschule steht auf Anfrage für Online-Besichtigungen zur Verfügung.

Danksagung Wir danken allen Teilnehmern des Kurses Mobile Security im Sommersemester 2021 an der Hochschule Schmalkalden. Besonderer Dank gilt Kevin Vondryska, Tim Trostmann, Marc Reitstetter, Tarlan Omarbayli und Ceyda Nur Gecimli für die engagierte Mitarbeit. Außerdem danken wir Fabian Fischer für seine hilfreichen Hinweise bei der Sicherheitsanalyse.

Funding Open Access funding enabled and organized by Projekt DEAL.

Literatur

1. Autonomous Networks Research Group (2016) Cooja Simulator. anrg.usc.edu/contiki/index.php?title=Cooja_Simulator&oldid=1730 (Erstellt: 21. Juli 2016). Zugegriffen: 10. Jan. 2022
2. BigBlueButton (2022) Bigbluebutton. bigbluebutton.org. Zugegriffen: 10. Jan. 2022
3. Contiki-NG Contiki-NG: The OS for Next Generation IoT Devices. github.com/contiki-ng/contiki-ng (Erstellt: 28. Jan. 2022). Zugegriffen: 31. Jan. 2022
4. Jitsi Meet. jitsi.org/jitsi-meet (Erstellt: 26. Jan. 2022). Zugegriffen: 10. Jan. 2022
5. libvirt (2022) libvirt Virtualization API. libvirt.org. Zugegriffen: 10. Jan. 2022
6. Lignan A Zolertia RE-Mote platform. https://github.com/Zolertia/Resources/wiki/RE-Mote/0418e5f500eda9614e9f20186a9783ed9fa99ab2 (Erstellt: 3. Nov. 2016). Zugegriffen: 19. Jan. 2022
7. noVNC (2020) Embedding and Deploying noVNC Application. github.com/novnc/noVNC/blob/master/docs/EMBEDDING.md (Erstellt: 20. Dez. 2020). Zugegriffen: 10. Jan. 2022
8. noVNC (2021) noVNC: HTML VNC Client Library and Application. github.com/novnc/noVNC/tree/v1.3.0 (Erstellt: 22. Okt. 2021). Zugegriffen: 10. Jan. 2022
9. QEMU (2021) Qemu. qemu.org (Erstellt: 23. Dez. 2021). Zugegriffen: 10. Jan. 2022
10. The Coding Machine Workadventure. workadventu.re. Zugegriffen: 10. Jan. 2022
11. Tiled Map Editor (2022) Tiled. mapeditor.org (Erstellt: 7. Jan. 2022). Zugegriffen: 10. Jan. 2022
12. websockify (2021) websockify: WebSockets support for any application/server. github.com/novnc/websockify. Zugegriffen: 10. Jan. 2022