applied sciences

MDPI

Article

A Case Study on the Design and Implementation of a Platform for Hand Rehabilitation

Tomaž Kosar [1,*], Zhenli Lu [2], Marjan Mernik [1], Marjan Horvat [1] and Matej Črepinšek [1]

[1] Faculty of Electrical Engineering and Computer Science, University of Maribor, 2000 Maribor, Slovenia; marjan.mernik@um.si (M.M.); marjan.horvat@gmail.com (M.H.); matej.crepinsek@um.si (M.Č.)
[2] Changshu Institute of Technology, Changshu 215500, China; zhenlilu@cslg.edu.cn
* Correspondence: tomaz.kosar@um.si; Tel.: +386-220-7448

Abstract: Rehabilitation aids help people with temporal or permanent disabilities during the rehabilitation process. However, these solutions are usually expensive and, consequently, inaccessible outside of professional medical institutions. Rapid advances in software development, Internet of Things (IoT), robotics, and additive manufacturing open up a way to affordable rehabilitation solutions, even to the general population. Imagine a rehabilitation aid constructed from accessible software and hardware with local production. Many obstacles exist to using such technology, starting with the development of unified software for custom-made devices. In this paper, we address open issues in designing rehabilitation aids by proposing an extensive rehabilitation platform. To demonstrate our concept, we developed a unique platform, RehabHand. The main idea is to use domain-specific language and code generation techniques to enable loosely coupled software and hardware solutions. The main advantage of such separation is support for modular and a higher abstraction level by enabling therapists to write rehabilitation exercises in natural, domain-specific terminology and share them with patients. The same platform provides a hardware-independent part that facilitates the integration of new rehabilitation devices. Experience in implementing RehabHand with three different rehabilitation devices confirms that such rehabilitation technology can be developed, and shows that implementing a hardware-independent rehabilitation platform might not be as challenging as expected.

Keywords: rehabilitation aid; movement observation; rehabilitation; assistive technology; robot-assisted rehabilitation; additive manufacturing; local production; human-computer interaction; domain-specific languages; code generation

check for updates

Citation: Kosar, T.; Lu, Z.; Mernik, M.; Horvat, M.; Črepinšek, M. A Case Study on the Design and Implementation of a Platform for Hand Rehabilitation. *Appl. Sci.* **2021**, *11*, 389. https://doi.org/10.3390/app11010389

Received: 11 December 2020
Accepted: 28 December 2020
Published: 3 January 2021

1. Introduction

Health systems worldwide face many emerging challenges, including population growth, population aging, and the prevalence of diseases. There is a substantial and increasing need for rehabilitation all over the world. However, specialized rehabilitation equipment is expensive. Rehabilitation institutions are confronted with the absence of rehabilitation facilities and equipment. Next, every rehabilitation department's prevalent limitation is the number of medical staff available for the rehabilitation service. Staff efficiency and service quality need to be addressed to reduce waiting times in rehabilitation centers. On the other hand, the question needs to be addressed if we are keeping up with modern times and including technology in the rehabilitation process [1–3]. The mentioned drawbacks (expensive equipment, lack of medical staff, lack of time to educate medical staff), can be reduced if computer technology takes the right place in the medical institutions. We believe modern technology (3D printing, micro-controllers, hardware customization, etc.) and advances in computer science (cloud solutions, open-source, etc.) can help to provide more efficient rehabilitation solutions [4–7] or even shift part of the rehabilitation process from medical institutions to the local or even home environment.

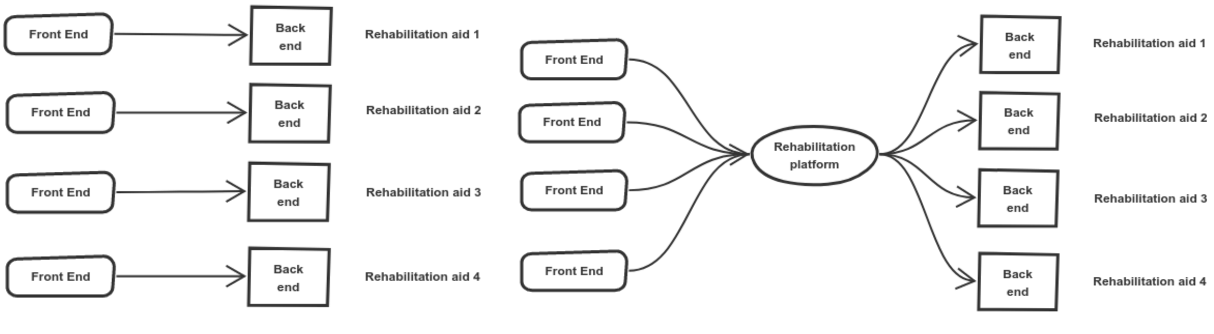
Technology, known under 3D printing and additive manufacturing [8], is the process of making a three-dimensional object from a 3D model with metal and plastic materials. There is already a wide range of 3D solutions available in medical research. Usually, they use 3D technology to reconstruct specific parts of the human body [9]. Recent advances in additive manufacturing bring tremendous promises for printing prosthetics for individual patient needs. These solutions are just coming into existence and beginning to display future potential [10]. 3D printed casts for a broken bone also speed up healing and improve the patient's life. In medical research, many challenges exist in additive manufacturing (integration with tissue, ethical standpoints, etc.). Assistive technology [11] concerns the technology which can improve the functional capabilities of persons with disabilities [4], and might make use of additive manufacturing easier than in medical research. Another technology that can help in the rehabilitation process is the consequence of the rapid development of robotics [12]. Robot-assisted rehabilitation training emerged in wide areas of rehabilitation [13], like hand functions [14], cerebral palsy [15], lower limb rehabilitation [16].

The number of rehabilitation tools in the market is growing. Some specialized devices serve particular treatments. On the other hand, there are multi-functional devices that serve a variety of purposes. Besides, tools change frequently—new hardware comes with new software, etc. As a result, the therapists need to learn new software each time they switch to new or upgrade existing rehabilitation devices. The ability to work with a unified system, thinking only on rehabilitation treatment, not dealing with specific settings of hardware or software would make work faster, simpler, and more intuitive for therapists.

This paper focuses on presenting the human-computer interaction system design and its implementation RehabHand, used by therapists to prepare customized tasks for the individual patients. In RehabHand, we use the benefits of Domain-Specific Languages (DSLs) [17–20]. DSLs are computer languages designed to solve tasks in a specific domain. In our case, the domain is a description of rehabilitation exercises, more specifically, hand rehabilitation exercises. One of the major benefits is end-user programming [21], meaning that domain experts write DSL programs without possessing knowledge of computer science and programming [22]. In our DSL, we used code generation technology [23] to enable the use of different hardware. On the market, you can find a wide range of robotic hands, exoskeleton hands, etc. The developed platform RehabHand demonstrates the use of different rehabilitation aids (even custom-made).

A fusion of the mentioned technologies, bearing in mind the equipment's total cost, and providing a customized rehabilitation aid for the individual patients, is our project's vision. The goal was to develop an aid for therapists who work within rehabilitation centers and engage patients in rehabilitation with modern solutions. The main contributions outlined in this paper are:

- the design of an affordable rehabilitation aid with cutting-edge technology that addresses some of the emerging challenges in health care and rehabilitation,
- a unique platform which enables therapists to define exercises independently of hardware and software (see Figure 1),
- a platform that can be upgraded with a new device (i.e., rehabilitation tool, robotic-assisted device, assistive technology) without changing the interface for the therapist,
- an example of local production for a rehabilitation aid where patients can work on their treatment autonomously to improve the functional capabilities with hand movement disabilities inside or outside rehabilitation centers,
- to support the rehabilitation community with a platform to share rehabilitation aid designs, rehabilitation exercises, and treatments.



(**a**) Traditional solutions (**b**) RehabHand

Figure 1. Comparison of RehabHand platform with traditional development and relationship between hardware and software.

The remainder of this paper is organized as follows—a background on hand rehabilitation challenges is given in Section 2. Section 3 discusses the motivation for our work. The architecture and other generalized ideas for the RehabHand application are described in Section 4. Results implementing and using RehabHand are described in Section 5. A discussion about our experience in implementing the rehabilitation platform follows in Section 6. The related work on rehabilitation is discussed in Section 7. Finally, the concluding remarks are summarized in Section 8.

2. Hand Rehabilitation Background

There are different physical disabilities [24]. In general, these are conditions that affect a person's mobility, stamina, or dexterity. There are many reasons for physical disability, like brain or spinal cord injuries, multiple sclerosis, cerebral palsy [25], and many more. Some may affect, either temporarily or permanently, a person's life. Examples of short term disabilities may include broken limbs, injuries, disabilities connected with surgery, medical treatments, or severe diseases. These patients experience various movement problems, and one of the main challenges is to reduce disability and raise their quality of life as quickly as possible. This process requires rehabilitation services to overcome these problems.

To improve strength, movement, or flexibility, disabled people need help from different therapists. Specific disabilities need special exercises in a clinical setting and experts' involvement (therapists, physiotherapists, occupational therapists, doctors, and others). In our paper, we address them superficially as therapists or medical staff. Each day millions of people with movement problems visit rehabilitation centers. Due to limited medical resources, patients need to wait for several weeks or months to start treatment [26]. Every delay in the start of rehabilitation has a negative impact on the rehabilitation result and rehabilitation time/cost. The advance of rehabilitation services [27] is one of the key factors in addressing the challenges of reducing waiting queues in rehabilitation centers worldwide.

Suppose we take an example of a hand and wrist temporary disability, which may happen due to surgery, fracture, dislocation, and so forth. Physiotherapy treatment starts with motion assessment of the hand, wrist, and fingers. Upon assessing finger flexion and extension deviation, therapists decide on a specific physical activity and provide exercises for them. Some exercises include training with therapists (passive range of motion); some include individual training (active range of motion). These exercises strengthen muscles, ligaments, and tendons around the joint, generate more power, or increase endurance. These exercises are helpful, for instance, for inflammation of the tendons.

The RehabHand platform addresses hand and finger individual training. There are two main user scenarios:

(a) A therapist prepares customized training for the individual patient, and supervise them occasionally in rehabilitation facilities.
(b) Individual training implemented at home or work sites with remotely guided and controlled rehabilitation.

During the design and implementation of the rehabilitation platform, we had these user scenarios in our minds.

3. The Motivation for RehabHand

There is a strong belief that, in the next decades, the human-computer interaction [28,29] in forthcoming systems will evolve and change permanently. We can already see many examples (gesture-controlled systems, speech, etc.). Here, one of the research topics is also to develop solutions that allow users who do not have a background in programming to develop their own applications [30]. This research topic is more known under End-User Programming or End-User Development (EUD) [21], supported with the usage of domain-specific language technology [17]. The idea behind EUD is to empower domain experts to express the desired functionality easily, without dealing with low-level software technologies (using frameworks, libraries, etc.), but develop programs with abstractions from a domain they know well. In this way we developed our platform RehabHand, to support therapists to write exercises in natural and domain-specific terminology. There are many benefits of such a development. On one hand, a therapist can prepare personalized exercises that are also readable for the patients. On the other hand, we foster the building of a community of domain experts who *speak the same language*, the language of RehabHand. Hence, they can share their exercises and related knowledge effectively with their colleagues. Since treatments are standardized, results can be compared between patients.

The other motivation for RehabHand is connected with the selection of devices for rehabilitation aid. Of course, the selection is driven by the specific treatment we would like to support. However, we can still be faced with the decision between several devices. We do not want to be tied to a single device; therefore, our idea was to prepare a platform which can be extended with a new rehabilitation hand. Many practical reasons exist for such independence from the hardware. If the currently used rehabilitation device does not cover our needs anymore, we can replace it with a different one. The up-front cost, level of effort, assumed risk, and latency required to make the software transition to adopt new hardware are minimized in our hardware-independent platform RehabHand. In Figure 1a traditional software-hardware relationship is compared to the RehabHand platform. In traditional development, manufacturers provide their own software for their devices. There is no connection between different software solutions. In RehabHand, a unified software is connected with a platform that is responsible for connecting different devices. As will be shown, RehabHand can be used, even if we custom develop our own RehabHand with 3D printing and use it instead of commercial solutions. RehabHand lowers the hardware adoption barrier, enabling software to achieve the benefits of RehabHand software customization with significantly less transition time and effort. Currently, as a proof of concept, our platform supports three different rehabilitation hands, but can be extended upon request. All three rehabilitation hands have minimal and elementary functionality, but there are groups of patients with associated disabilities—besides hand movement, they also have hearing, vision, spatial awareness, and so forth. Although our examples are rather simple rehabilitation tools, such rehabilitation aids represent indispensable assistance in the rehabilitation process.

The next, but not least, motivations for RehabHand are the possibilities opened with such a platform. Recall the second usage scenario from Section 2, using a rehabilitation aid at home. Since the material for 3D printing, microcontrollers, micro-servos and other electronic elements are affordable, we can construct a rehabilitation tool with local production (upon given design instructions). This ability provides a solution for many emerging challenges in the rehabilitation we are dealing with. Imagine the time and effort of the elderly needed to come to rehabilitation centers. Instead, rehabilitation platforms like RehabHand enable therapists to connect with patients over the internet and control rehabilitation aid remotely. Imagine helping someone in low or middle-income countries without proper rehabilitation facilities, or someone with movement problems not able to visit the rehabilitation center. Apart from this, therapists need to save exercises, observe previous treatments, and similar functionalities for dislocated patients. Today these functionalities can be achieved with cloud computing. We believe such technology can provide a way to easier access to rehabilitation services. Besides, it promotes well-being for all ages.

4. Method: A Design of an Extensive Rehabilitation Platform

During RehabHand implementation, we came up with many ideas about what a modern rehabilitation aid should contain. The top-level of the propaedeutic rehabilitation platform is presented in Figure 2. We envision five types of stakeholders in an extensive rehabilitation platform—therapists, patients, communication engineers, rehabilitation aid engineers, and the community. The architecture consists of 3 layers—a software layer, a communication layer, and a hardware layer.

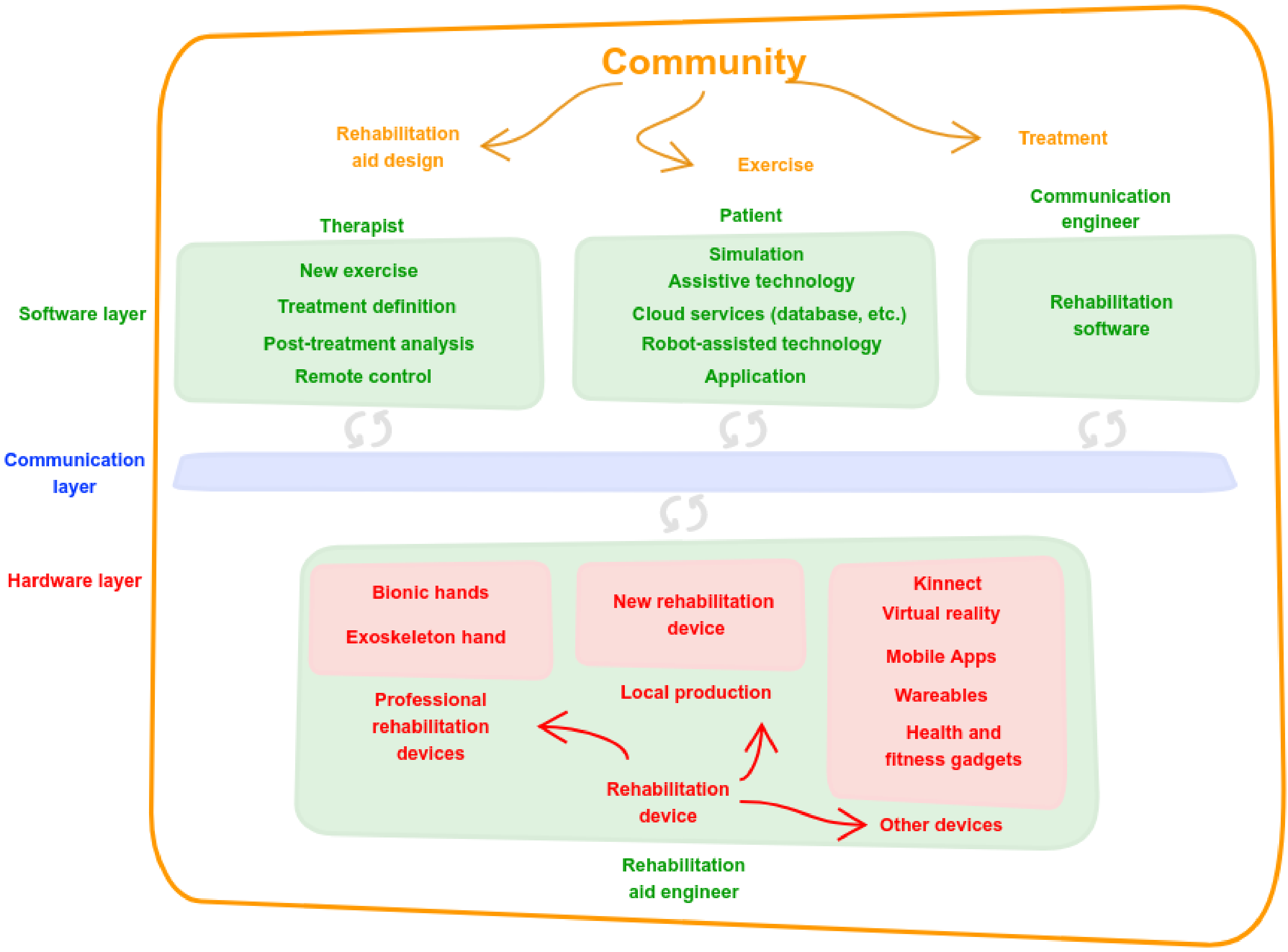


Figure 2. Extensive rehabilitation platform.

In the software layer, therapists are responsible for the definition of new exercises. They also define the treatment plan, which basically means selecting and customizing already defined exercises for a specific patient. If the rehabilitation aid can return feedback from treatment [31], a post-treatment analysis [32] should be supported in the software layer. As shown further in the paper, remote-control [33] is another service that we envision with high priority in future rehabilitation aids. The software layer should be independent—meaning, no matter which rehabilitation device the patient is using, the definition of an exercise should not change.

The software layer of an extensive rehabilitation tool should also provide support for patients. Due to different patients' needs, software support can be of various forms and solutions. Note that a combination of solutions can also help some rehabilitation cases (robot-assisted technology together with guidance over speech [34]).

The communication layer assists as a glue between the software and hardware layer. It serves as an interface between rehabilitation knowledge (defined within exercises and treatments) and all supported devices. Each exercise needs to be translated into computer instructions for every supported device. The communication layer is the communication engineer's responsibility. In the case of our platform, we name this person RehabHand engineer. This is discussed in further Section 5.2.

In the hardware layer, we envision professional rehabilitation devices, local production devices, and other devices. Professional rehabilitation devices that are special-purpose devices and are bought on the market. These devices can come with dedicated software, but to be included in the platform, they need to come at least with an Application Programming Interface (API) that we can use to connect with the software layer. In the hardware layer, an extensive rehabilitation platform should support rehabilitation devices constructed from accessible hardware with local production and other devices, which we can use as a rehabilitation aid—simulations in the form of Virtual Reality [6] and devices like Kinect [35] have already proven their usefulness for the rehabilitation process. Mobile phones, bands, health and fitness gadgets, have many sensors that can be useful during rehabilitation, and are accessible over the Internet for monitoring patients' data. Anyone constructing a modern rehabilitation aid should think which of these devices should be supported in their forthcoming product.

Soon, we believe local production will be an important part of modern rehabilitation aids. Combining additive manufacturing with affordable electronics, we can construct customized rehabilitation aids. We name this engineer a Rehabilitation Aid Engineer, a person who constructs the rehabilitation aid. In the field of Digital Fabrication [36], these engineers call themselves *Makers*—anyone producing needed objects. Awareness of the importance of health rises. A new role of the patient, medical staff, the patient's family, or anyone who can become an active participant in creating customized rehabilitation aids. There are already strong communities in rehabilitation (sharing designs of assistive technologies, robot-assisted tools, etc.). In the future, we envision other active participants of the community who share exercises and treatments between users in a unified form, independent of a device.

Note, we refer with rehabilitation aids to software, different devices, such as rehabilitation tools, assistive technologies, robot-assisted tools, and so forth. More specifically, all rehabilitation support with the help of IT technology.

RehabHand: An Example of an Extensive Rehabilitation Platform

In this paper, we will concentrate on the communication layer mentioned above. We believe it needs to be described formally, close to domain terminology, and support automation (verification, validation, and tool generation). This can be achieved with the use of DSLs instead of using classical APIs or services. We developed RehabHand, which is based on language processing technology, as shown in Figure 3. The medical staff writes a new exercise in the form of a program in the language RehabHand. This program represents customized exercises for an individual patient. The DSL program is then taken by the parser, which checks the exercise structure. The next step in the language processing algorithm is the code generation, that generates the target device's program. The RehabHand application then ensures that the translated code is uploaded to the target device. The last step is that the rehabilitation application starts the communication with the rehabilitation device (for example, the Arduino-based rehabilitation hand).

All the alternatives in the domain of Hand Rehabilitation are combined inside the DSL, and a specific hand exercise is actually defined with a programming language program. We believe preparing exercises in such a manner is easy, transparent, and unambiguous. Such a program is used as input in the communication layer to connect with a specific rehabilitation device. As will be shown in our examples, the exercise defined with the language RehabHand is intuitive. We used the code generation technique's benefits and compiled our code to three different devices. We have two professional rehabilitation devices: Bionic arm (bionic arm manipulator connected with the Arduino Mega 2560 bought on https://world.taobao.com/), and Cyton (Robot hand produced by the Robai company, http://new.robai.com/) [37]. As a third rehabilitation hand, we decided to create a custom-made rehabilitation aid, an Arduino-based device. Most of the hand was 3D printed. This rehabilitation hand was used as a proof of concept and a testing platform during the RehabHand application development.

Other parts of the communication layer are described just briefly in the paper. One important part is RehabHandInterpret, which enables medical staff to select a list of exercises a patient needs to perform during a rehabilitation treatment. The order of exercises, number of repetitions, and so forth, are selected before being uploaded to specific hardware. These functionalities' main benefit is that all exercises for an individual patient are uploaded simultaneously to the rehabilitation hand. Application controls which exercise is the next to be performed by a patient.

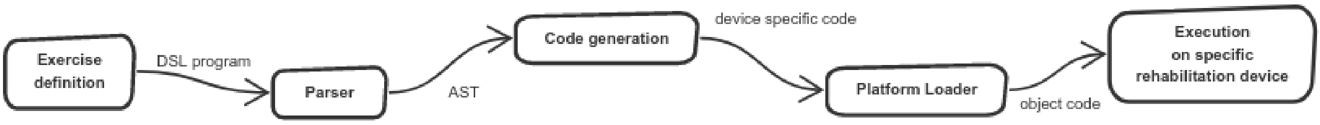


Figure 3. Language processing in RehabHand.

5. Results

To understand the above architecture, we will outline three user viewpoints in the following sections: Therapist, RehabHand engineer, and Patient viewpoint.

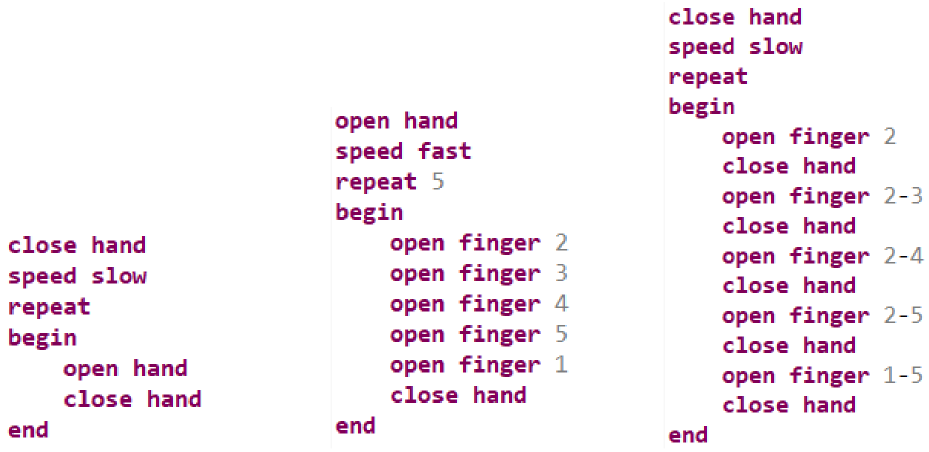
5.1. Therapist Viewpoint

Although the work's essence is in presenting the rehabilitation platform's design, one of the crucial parts are the exercises written inside RehabHand. A therapist can write an arbitrary rehabilitation exercise easily in natural notation.

Figure 4 shows three simple exercises written with RehabHand. Albeit Hand Rehabilitation is a small domain, we have numerous combinations for constructing hand rehabilitation exercises. The first program (Figure 4a) allows the patient to repeat the opening and closing of the hand slowly. This strengthens the muscles of the entire wrist. The following program (Figure 4b) shows the exercise with the fingers' sequential lifting. We start with the index finger and finally finish with the thumb. When all the fingers are lifted, the patient needs to close the whole hand. The third program (Figure 4c) shows the lifting of a combination of fingers. First, we lift the index finger and close it, lift the index and middle fingers simultaneously, close them, and so on. With the RehabHand platform, the therapist can prepare exercises for an individual patient based on their specific needs.

We can observe the simplicity of RehabHand notation. Actually, no additional specific knowledge is required to write exercises. Therapists write simple commands which they would normally speak to a patient. They can concentrate on motion disabilities for a specific patient and write appropriate, customized exercises. Thus, no programming knowledge is needed to write rehabilitation programs.

Traditional human-computer interactions using Graphical User Interfaces (GUIs) are subject to frequent changes. For every version or release of an advanced device, a GUI is changed to support features behind the scene. It is the same with software for rehabilitation aids, assistive technology, and robot technology. Consequently, therapists have to adapt each time they switch to a newer rehabilitation device or change the assistive technology provider. In RehabHand, if we support new features or change hardware, the language's basic notation will stay the same. Therapists' main concern is exercise definition, and the exercise will work no matter which hardware or software is used. These tools (compared with the RehabHand), are limited to one type of hardware manufacturer, while RehabHand can support different hardware types from different manufacturers.



```
close hand
speed slow
repeat
begin
    open hand
    close hand
end
```

(**a**) Open-close hand

```
open hand
speed fast
repeat 5
begin
    open finger 2
    open finger 3
    open finger 4
    open finger 5
    open finger 1
    close hand
end
```

(**b**) Individual finger

```
close hand
speed slow
repeat
begin
    open finger 2
    close hand
    open finger 2-3
    close hand
    open finger 2-4
    close hand
    open finger 2-5
    close hand
    open finger 1-5
    close hand
end
```

(**c**) Finger combination

Figure 4. Different exercises for hand rehabilitation.

5.2. RehabHand Engineer Viewpoint

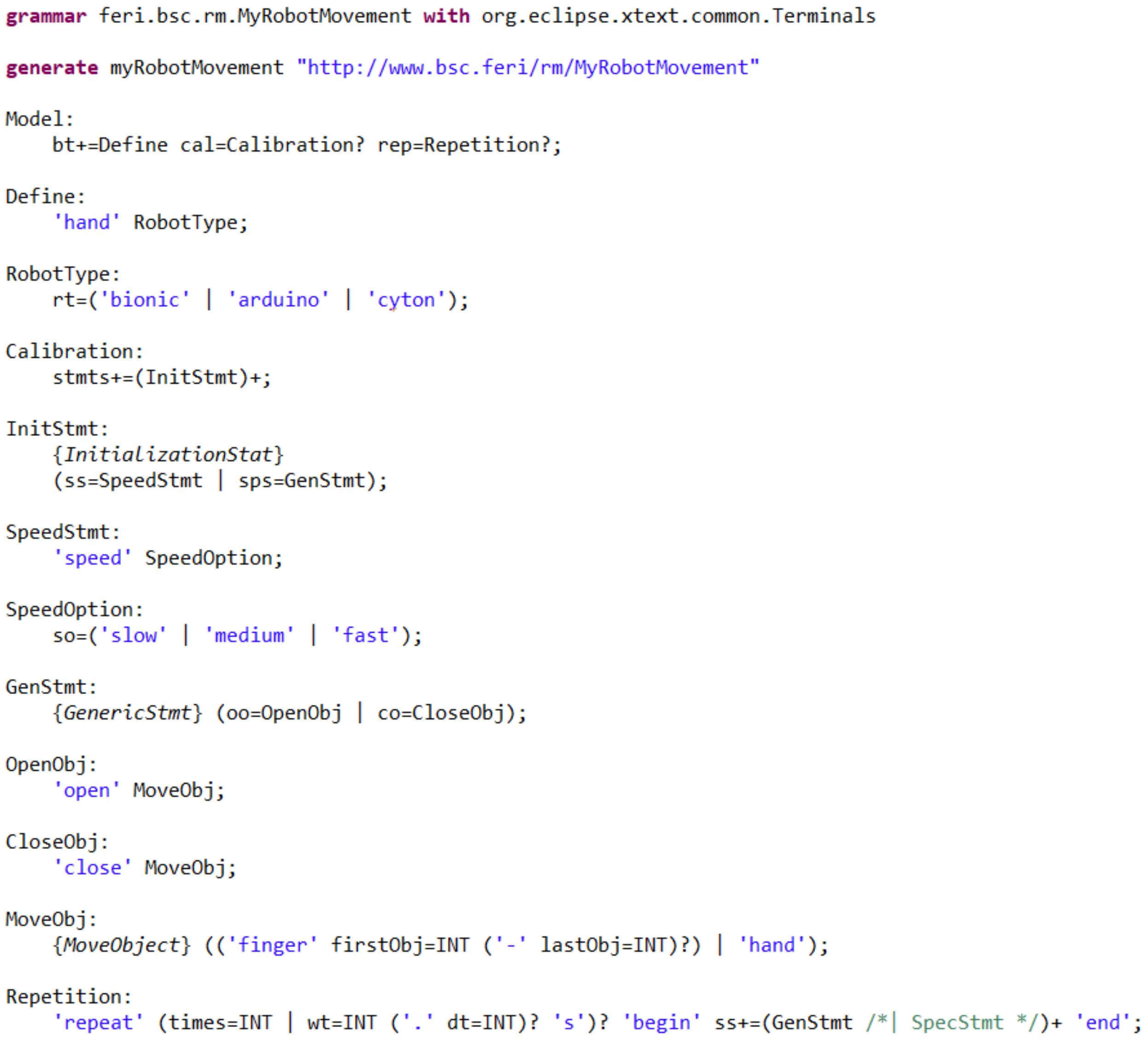
We need an engineer who defines the communication protocol between the various stakeholders in the rehabilitation service development. In our case, due to the high abstraction, this is already DSL. DSL allows us to automate the support of various devices. This part is verified but, unfortunately, less known. Therefore we explain it on the example of RehabHand in detail.

The following Subsection explains the RehabHand implementation that parses exercises from Figure 4. The design of the language RehabHand is described with the syntax (Figure 5) and code generation (Figure 6) to different rehabilitation aids.

5.2.1. Domain-Specific Syntax

Domain-Specific Languages [38,39] are languages tailored to a specific application domain [17,40]. We often call them little languages [41], but it does not always have to be the case. DSL programs are expressive, short, and easy to maintain [42]. Several techniques exist for the implementation of a Domain-Specific Language [43]. Besides the embedded approach [44], the most commonly used approach is compiler/interpreter [19]. There are notable advantages with this approach [43]. The most important one is syntax, which can be prepared without any restrictions. The compiler generator approach is related to the compiler/interpreter approach. Here, some phases are generated automatically (lexical and syntax parts) using language development systems (compiler-compilers) (e.g., ANTRL [45], Lex/Yacc [46], LISA [47], YAJCo [48]). A similar approach to compiler generators is language workbenches (Xtext in Eclipse, or Jetbrains MPS in IntelliJ), a term coined by Martin Fawler [22]. They contain a set of tools that support the development of a Domain-Specific Language. The most commonly used feature is to support an integrated development environment (IDE) for DSLs' end-users.

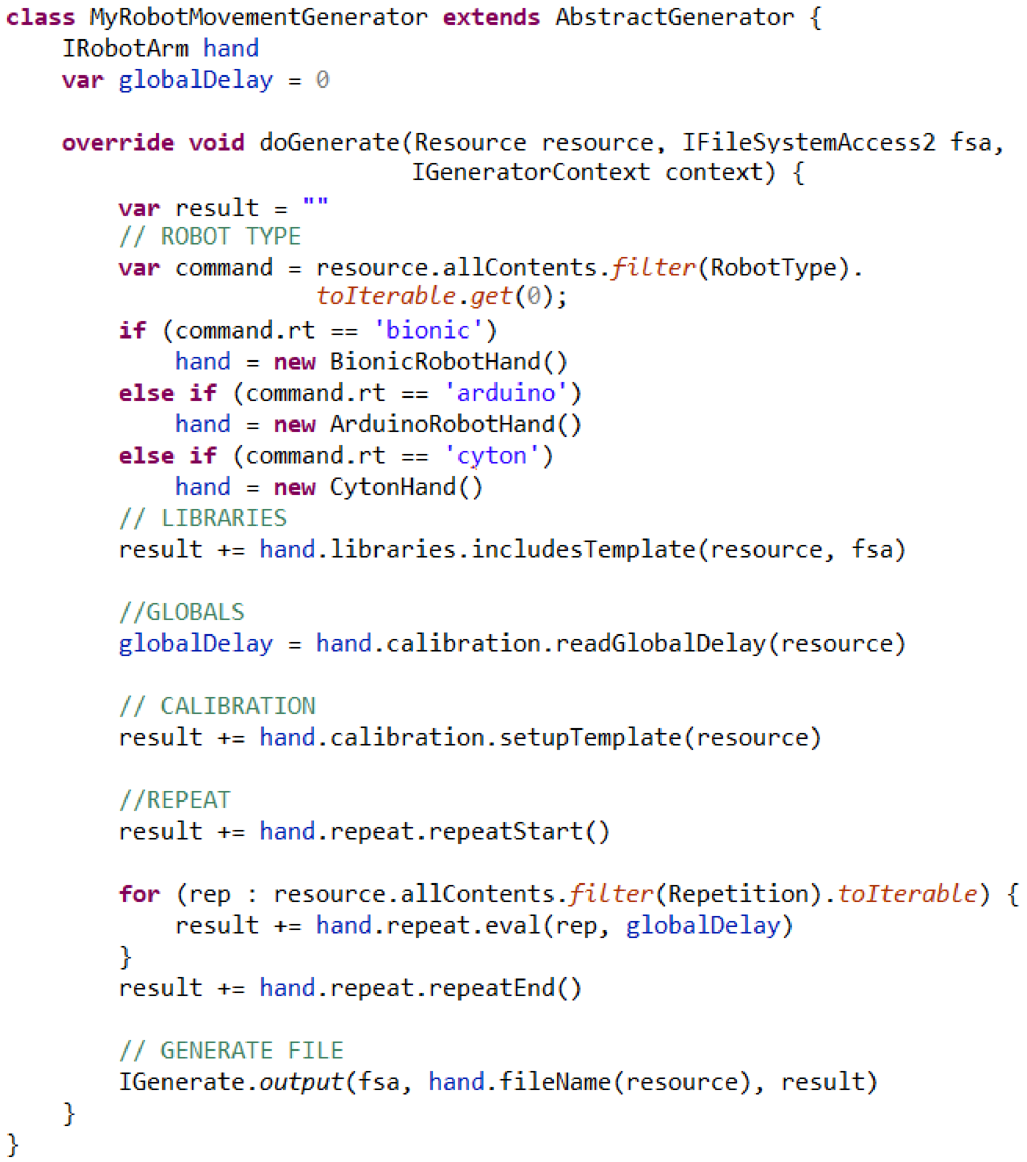
RehabHand syntax was developed in the language workbench Xtext [49]. Xtext is implemented in Java and is available in the form of a plugin for the IDE Eclipse. To define a DSL, we first define a grammar in an EBNF-like syntax. Based on the grammar, the Eclipse plugin will then generate a compiler (using compiler generator ANTLR [45] beneath) and the language infrastructure support. This generation process, as well as the resulting tools, can be customized in numerous ways (constraints on the syntax tree are implemented by *validators*, syntax tree elements with *label providers*, etc.).



```
grammar feri.bsc.rm.MyRobotMovement with org.eclipse.xtext.common.Terminals

generate myRobotMovement "http://www.bsc.feri/rm/MyRobotMovement"

Model:
    bt+=Define cal=Calibration? rep=Repetition?;

Define:
    'hand' RobotType;

RobotType:
    rt=('bionic' | 'arduino' | 'cyton');

Calibration:
    stmts+=(InitStmt)+;

InitStmt:
    {InitializationStat}
    (ss=SpeedStmt | sps=GenStmt);

SpeedStmt:
    'speed' SpeedOption;

SpeedOption:
    so=('slow' | 'medium' | 'fast');

GenStmt:
    {GenericStmt} (oo=OpenObj | co=CloseObj);

OpenObj:
    'open' MoveObj;

CloseObj:
    'close' MoveObj;

MoveObj:
    {MoveObject} (('finger' firstObj=INT ('-' lastObj=INT)?) | 'hand');

Repetition:
    'repeat' (times=INT | wt=INT ('.' dt=INT)? 's')? 'begin' ss+=(GenStmt /*| SpecStmt */)+ 'end';
```

Figure 5. RehabHand grammar in Xtext notation.

A simplified version of RehabHand Xtext syntax is presented in Figure 5. There are various non-terminals derived from the starting non-terminal `Model`: `Define`, `Calibration`, and `Repetition`. `Define` is mandatory, `Calibration` and `Repetition` are optional. Non-terminal `Define` is further decribed by terminal `hand` and non-terminal `RobotType`. Non-terminal `RobotType` defines currently supported rehabilitation hands (terminals `bionic`, `arduino`, and `cyton`). Note that there is no hand definition (e.g., `hand arduino`) in the exercises in Figure 4. This is not part of the rehabilitation exercise. Therefore, we decided to omit any system configuration from the domain definition. This line is added in the preprocessing of exercises in Figure 4. If the `Calibration` is desired within a RehabHand program, initialization is performed (non-terminal `InitStmt`). Non-terminal `InitStmt` is defined with speed (non-terminal `SpeedStmt`) or hand initialization position (non-terminal `GenStmt`). Later is defined with the opening command (non-terminal `OpenObj`) or closing command (non-terminal `CloseObj`). Both non-terminals `OpenObj` and `CloseObj` are associated with non-terminal `MoveObj`, that basically provides the possibility to work with a finger or the whole hand. After calibration, the repetition process starts the rehabilitation exercise (non-terminal `Repetition`). Repetition can be defined with a number of repetitions, timed or omitted, which basically means the exercise will go on until canceled by the end-user. Between terminals `begin` and `end`, the `GenStmt` is executed at least once (operator '+' following non-terminal `GenStmt`).



```
class MyRobotMovementGenerator extends AbstractGenerator {
    IRobotArm hand
    var globalDelay = 0

    override void doGenerate(Resource resource, IFileSystemAccess2 fsa,
                             IGeneratorContext context) {
        var result = ""
        // ROBOT TYPE
        var command = resource.allContents.filter(RobotType).
                      toIterable.get(0);
        if (command.rt == 'bionic')
            hand = new BionicRobotHand()
        else if (command.rt == 'arduino')
            hand = new ArduinoRobotHand()
        else if (command.rt == 'cyton')
            hand = new CytonHand()
        // LIBRARIES
        result += hand.libraries.includesTemplate(resource, fsa)

        //GLOBALS
        globalDelay = hand.calibration.readGlobalDelay(resource)

        // CALIBRATION
        result += hand.calibration.setupTemplate(resource)

        //REPEAT
        result += hand.repeat.repeatStart()

        for (rep : resource.allContents.filter(Repetition).toIterable) {
            result += hand.repeat.eval(rep, globalDelay)
        }
        result += hand.repeat.repeatEnd()

        // GENERATE FILE
        IGenerate.output(fsa, hand.fileName(resource), result)
    }
}
```

Figure 6. Generation of the code.

From the syntax defined in Figure 5, a lexer and parser are built automatically. Without using Xtext or a similar language workbench and compiler generators, we still have the opportunity to write them by hand [43].

For the sake of simplicity, in the language in Figure 5, we only have rehabilitation exercises related to opening and closing the whole hand and fingers. However, as shown in Figure 4, there are plenty of possibilities, even with such a small language. The grammar from Figure 5 can be expanded if we want additional movements, or to move individual finger joints, prepare extension and flexion rehabilitation exercises. Observe the commented non-terminal `SpecStmt` as an entry point to a more detailed definition of hand and finger motions.

Note that exercise possibilities are connected with the rehabilitation hand we use. The rehabilitation hand will not make the desired move if the functionality is not available for that device. We report this to the therapist who is constructing an exercise for a specific patient and rehabilitation device.

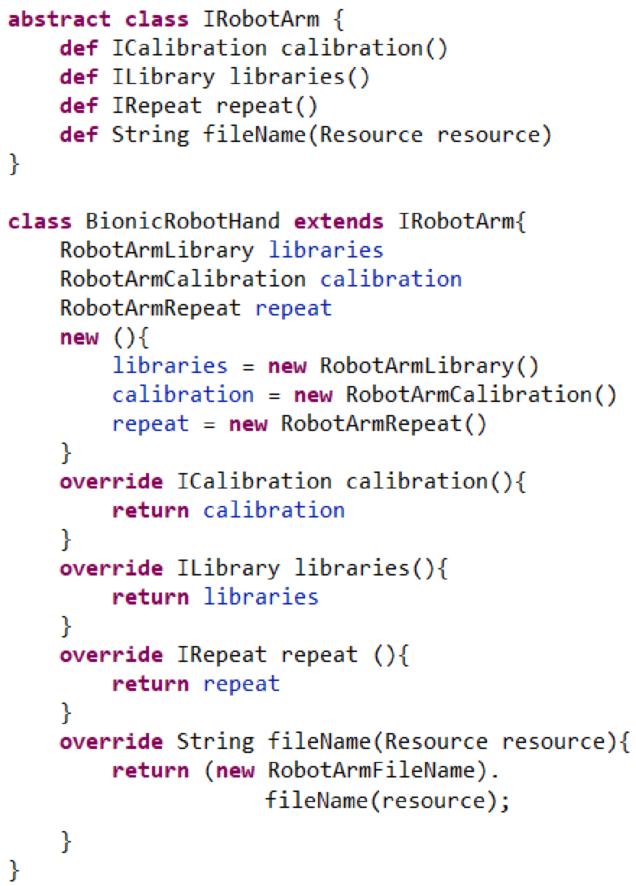
5.2.2. Code Generation

The semantics of the RehabHand is described using the language Xtend [49]. Xtend is a general-purpose programming language for the Java Virtual Machine. More specifically, Xtend is a dialect of Java programming language but has a more concise syntax. Any Java library can be used with Xtend. Usually, Xtend is used for code generation of a DSL developed with Xtext.

Translators from RehabHand notation to three different devices were defined from Xtend specifications in Figure 6 inside *if-else* statements (method `doGenerate`) which create variable `hand` with help of three different classes: `BionicRobotHand`, `ArduinoRobotHand`, and `CytonHand`.

In the semantics of the `doGenerate` method, an attribute `result` carries the translated program through this method and finally outputs it in the file (see the last statement of the `DoGenerate()` method). Translated code in the attribute `result` is constructed during processing the DSL program. We read any library includes, calibration, and evaluate the repetition of actions with hand/fingers from the DSL program (see *for* loop inside the doGenerate() method).

Figure 7 presents a high-level part of the Xtend code to a support specific rehabilitation hand. The object model of any code that controls the rehabilitation device needs to correspond to the abstract class IRobotArm, where basic functions are defined. Variable `hand` corresponds to this abstract class (see Figure 6, again).



```
abstract class IRobotArm {
    def ICalibration calibration()
    def ILibrary libraries()
    def IRepeat repeat()
    def String fileName(Resource resource)
}

class BionicRobotHand extends IRobotArm{
    RobotArmLibrary libraries
    RobotArmCalibration calibration
    RobotArmRepeat repeat
    new (){
        libraries = new RobotArmLibrary()
        calibration = new RobotArmCalibration()
        repeat = new RobotArmRepeat()
    }
    override ICalibration calibration(){
        return calibration
    }
    override ILibrary libraries(){
        return libraries
    }
    override IRepeat repeat (){
        return repeat
    }
    override String fileName(Resource resource){
        return (new RobotArmFileName).
                    fileName(resource);

    }
}
```

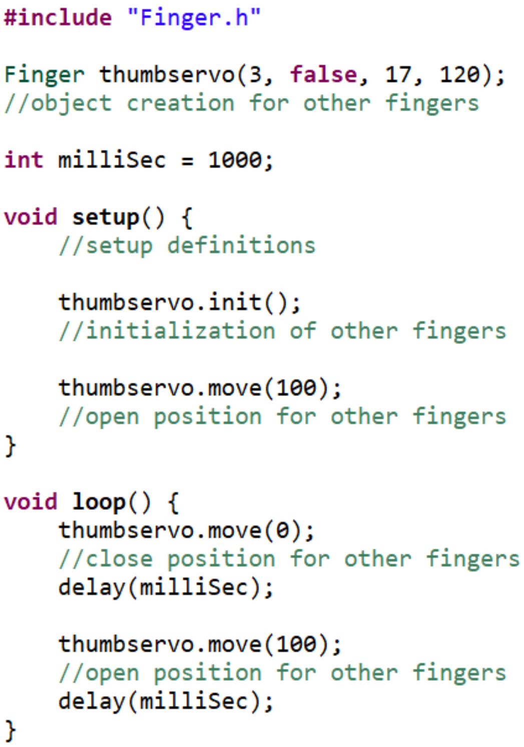
Figure 7. Top level implementation of Bionic rehabilitation hand code generation.

Implementation of a bionic hand with the concrete class `BionicRobotHand` (Figure 7) corresponds to this abstract class. This class implements all abstract methods (`calibration()`, `libraries()`, `repeat()`, and `fileName()`) of its super abstract class IRobotArm. Very generally speaking, the idea behind these methods is to provide part of the generated code returned for a specific rehabilitation hand.

5.2.3. Presentation of Generated Code

Amended generated code of the exercise in Figure 4a is presented in Figure 8. Because of the actual Arduino code's length, some parts were omitted and replaced with descriptive comments.

Arduino code is a C++ code with some additions. Generated code is always a sum of static and dynamic code. No matter what exercise we define, static code always appears in the generated code. In the case of RehabHand, the static part of the generated code was the inclusion of library (i.e., `Finger.h`), global variables for each finger (`thumbservo`), and other useful variables that control the sequence of exercise execution (number of repetitions, motion delay, etc.). Variables representing fingers are instantiated with several parameters (Arduino port number, option reverse, and motion range—min and max position of a servo). Each Arduino program constitutes of function setup() and loop(). The setup() function is called when the Arduino program starts, and is responsible for initializing different parts of device functions. It only runs once, after each start or reset of the Arduino board. Our function setup() is responsible for initializing servos connected with fingers (`thumbservo.init()`), and so forth.



```
#include "Finger.h"

Finger thumbservo(3, false, 17, 120);
//object creation for other fingers

int milliSec = 1000;

void setup() {
    //setup definitions

    thumbservo.init();
    //initialization of other fingers

    thumbservo.move(100);
    //open position for other fingers
}

void loop() {
    thumbservo.move(0);
    //close position for other fingers
    delay(milliSec);

    thumbservo.move(100);
    //open position for other fingers
    delay(milliSec);
}
```

Figure 8. Generated code for Arduino platform.

The dynamic part starts with setting fingers at the close hand position (`thumbservo.move(100)`). This generated code corresponds to the first line of the program in Figure 4a (`close hand`). The rest of the dynamic code is contained in the `loop()` function, which repeats infinitely—when `loop()` function execution is completed, it starts again from the beginning. Observe that code generation may not always be one to one mapping between exercise definition to the generated Arduino code. Line `speed slow` from Figure 4a is a typical example. It corresponds to both `delay` statements in the `loop()` function, as well as global variable definition (`milliSec`).

The library from the beginning of the generated code in Figure 8 simplifies finger manipulation of a hand. Our library is adapted for the servo motors that we use. This library needs to be updated for different actuators (stepper motors) and the rehabilitation hands we use.

Note that the generated source code is usually not formatted. In Figure 8 we inserted indentation in the code for presentation purpose. The generated code is not intended to be read by programmers. It is uploaded, build, and run automatically on an Arduino device.

5.3. Patient Viewpoint

In the following Subsection, we provide several use-cases of why such a design of rehabilitation aid is useful and what advantages we gain using our ideas.

In rehabilitation centers such as Changshu No.2 People's Hospital in Suzhou City (Jiangsu Province, P.R.China), it is always a problem to take care of all who need rehabilitation therapy. The constant shortage of staff is not only a problem in Changshu hospital; similar rehabilitation centers face this problem elsewhere in the world. Rehabilitation aids with similar effect as RehabHand (Kinect, VR, etc.) provide an opportunity that therapists can treat several patients simultaneously. Given exercises to rehabilitation aids, patients can work autonomously to a limited extent, while therapists supervise, observe, and guide several patients through the treatment simultaneously.

Besides, during our pilot study, therapists confirmed that the rehabilitation hand is a useful aid during the rehabilitation process and reduces their effort. Note that the therapist already uses several tools during the treatment. Some are very simple, for instance, a squeeze ball. A ball to squeeze into the palm with fingers can be used to strengthen hand and finger grip. Patients need to squeeze the hand as hard as possible and hold it there for a few seconds. Patients need to repeat this several times (for instance, 10 times, it depends on the injury, age, power). Another similar exercise with the squeeze ball is pinch strength. Patients pinch the ball between finger and thumb tips and hold it for a few seconds (i.e., 60 s). Again, patients do the repetition with this exercise. These simple tests need to be done with caution to prevent further damage. Pace, number of repetitions, and so forth, need to be controlled by the therapist. In our case, we used a bionic arm to define the guidelines for a patient (see it in Figure 9 between the patient and the therapist). Patients observe the bionic hand and repeat as indicated.

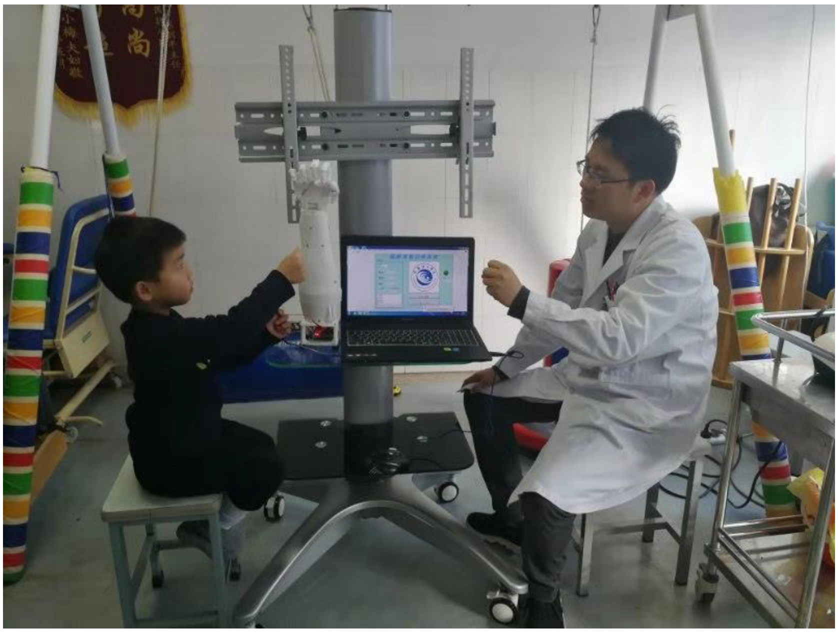


Figure 9. Bionic arm with RehabHand software usage in action.

The implementation example in Figure 10 shows a rehabilitation hand, which we made during the creation of the RehabHand platform with local production, and used for our demonstration purpose. The idea was to make a simple (to make and to use) robotic hand that is self-explanatory and intuitive for the patient. Another criterion was simple electronics. A simple and fast 3D model for production that does not require high tolerances is the next advantage. Each finger is controlled with one actuator. Complexity increases exponentially with each additional degree of moving space. Our rehabilitation hand uses small servo motors that do not require additional motor drivers. The size of the rehabilitation hand is adjusted to accommodate the motors. Arduino is used for controlling motors. Specifically, Arduino Nano was chosen because of its small size. The programs in Figure 4 were prepared for this rehabilitation hand.

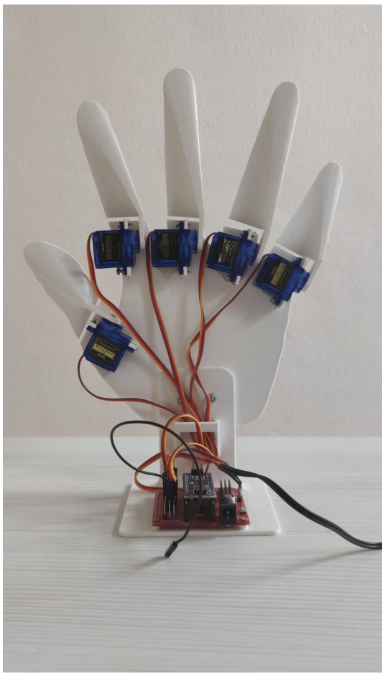


Figure 10. Arduino based rehabilitation hand.

Finding new innovative solutions for rehabilitation is inevitable, and some argue it is even good to engage patients in the process of co-designing assistive technologies [50–53]. The participant became engaged in a personal innovative process, starting to deploy everyday objects as assistive devices. Our study contributes to Reference [50]; we believe local production with 3D printing of specific parts of rehabilitation aids opens up a way to home usage and affordable rehabilitation solutions. We envision a rehabilitation aid constructed from accessible software and hardware with local production, and want to give a solid ground with our architecture and design to future rehabilitation aids. Our design can be reused, not only for rehabilitation aids, but for many assistive technologies.

As already mentioned in Section 4, a rehabilitation hand is one example of patients' interaction systems. Exercises written with Domain-Specific Language can serve as an intermediate layer for several services or devices needed in the rehabilitation process. One of them is the text-to-speech conversion of the exercises written in Figure 4. Although programs are written in English, we can localize terminal symbols from the Xtext specification (see Figure 5, again) to support any other language. During our project, we have already tried to support the Chinese language for partners in a project. Besides the rehabilitation hand, one could use Virtual Reality for guiding rehabilitation, and other motivating rehabilitation aids, like Kinect [35].

The transparent and modular RehabHand platform allows other useful usage scenarios. Imagine a patient from a lower-income country (with basic access to technology but a lack of advanced resources) having the hand motion disorders described several times in the paper. With local production, anyone can help at producing a rehabilitation aid for this patient. The donor can send a customized, affordable rehabilitation hand from anywhere in the world. Still, this patient can not use the rehabilitation aid on his own. He needs a guide during usage. Using an Internet connection therapists can guide him during treatments. Moreover, the therapist can even control such a device remotely, and send customized exercises to the rehabilitation aid. We control our rehabilitation aid remotely using an Android mobile application, sending different exercises to the device, and so forth. Robots' remote-control in medicine, even in surgery [54], is a trendy research field. We believe a rehabilitation aid might use remote-control for those who are unable to visit rehabilitation facilities. One of our further research activities in rehabilitation aid will also be focused on this area.

6. Discussion

In the following Section, we discuss our experience developing a rehabilitation aid platform and having a pilot study in a rehabilitation institution. Furthermore, we highlight some factors that will help future rehabilitation aid developers and engage the community.

6.1. Benefits and Obstacles for Using Local Production Rehabilitation Aids in Rehabilitation Facilities

Global health care costs were rated at US$7 trillion in 2015, but in 2020, the estimated global healthcare costs were estimated at US$8.7 trillion [55]. However, this estimation was calculated before COVID-19 appeared in the early months of 2020. The same trends are emerging with rehabilitation costs. The highest absolute growth of physical rehabilitation needs was observed in lower-income countries that usually have deprived rehabilitation services [55]. Finally, a significant reduction in age-standardized needs per capita occur only in high-income countries, which can be related to the wider implementation of physical rehabilitation services in these countries. To reduce these costs further, health care institutions must adapt and seek new opportunities continuously. Technological advances can be of great support. The RehabHand is one such example of an advanced rehabilitation platform that can reduce waiting queues in rehabilitation centers. Therapists do much repetitive work with patients, which could be reduced using modern rehabilitation tools and doing treatments with several patients simultaneously.

We believe future rehabilitation aid manufacturers should think about functionalities, devices, and unified description of exercises, as discussed in Section 4. We are aware that RehabHand is not a panacea for open issues in rehabilitation institutions, or that rehabilitation institutions' Regulations and Standards would support the use of a rehabilitation aid supplemented with local production. Therefore, we foresee many complications in applying such a platform in modern health care. A potential rehabilitation aid manufacturer needs to comply with regulations, safety, sustainability issues, and so forth to enter rehabilitation facilities. It would be easier for the RehabHand platform to come to life in a community where active participants seek rehabilitation aids that would enable part of the rehabilitation process in the home environment.

6.2. Affordable Rehabilitation Aids

Most rehabilitation aids that do most of the job in the treatment process are affordable solutions. Imagine a squeeze ball again, as an example. We follow this philosophy with our vision of a extensive rehabilitation platform where we advocate devices and technology which are already available on the market (Kinect, wearables) and local production.

Most innovations never cross the border of becoming commercialized [50]. Sharing those designs with the community could be useful for the rehabilitation of other patients. People with rehabilitation needs may speed up the rehabilitation process by using publicly available and affordable solutions.

6.3. The Role of Local Production

When choosing the right robotic arm for the project, many possibilities have to be considered. Finding a rehabilitation hand having all the desired requirements is less likely. Choosing the one that suits all requirements is not a simple task, but there is also the possibility of taking all desired features and merging them into a new product. There are three options to choose from:

(a) Buy a professional rehabilitation device, plug it in, and start to use it.
(b) Choose a design (get it from community sources) and make it yourself.
(c) Make your own design.

The bare minimum of requirements for the robotic arm should be known. Price is the most common limitation, but it is, by far, not the only one. Different options require different tools and skills:

(a) Assembled robotic hands that come in a box are ready to use and have been tested. That is the fastest and the simplest solution. The main drawbacks are costs, and to which degree the rehabilitation device matches the requirements of rehabilitation. In case of a problem, you can contact the manufacturer for more information.
(b) Many designed and prepared projects for doing it yourself lack reliability, and they work under limited workloads. You have to have a 3D printer, and have to produce assembling parts by yourself. 3D printers produce parts with larger tolerance. That can be a problem when parts need to press fit. Printing settings have to be adjusted according to the 3D printer used. This requires many trials. Required electronic and mechanical components are sometimes sold together in kits. If they are not, each part has to be ordered separately. Finding the right one can represent a challenge. In the end, everything needs to be assembled. In the case of malfunctioning, you have to be able to solve it by yourself.
(c) Third, and the most demanding option, is to make the entire design for a rehabilitation hand by yourself. Different knowledge within different fields is required (production from design till manufacturing, electronics, programming, etc.). The path to the final product takes time. Also, other hidden costs have to be considered (malfunctioning, maintenance, upgrade). The main advantage is a custom-made rehabilitation aid for a reasonable price, and sometimes a custom made part is the only option.

How does this decision affect the RehabHand platform? If we have chosen supported devices in the case of (a) and (b), the code generation part in the glue communication layer is already provided. In the case of (c), we need to contact the RehabHand developers to provide the code generation.

With the use of additive manufacturing, together with appropriate electronics, you can get a long way. 3D printers get cheaper, do not require much space, and also do not require much knowledge to operate. Companies use them for prototyping or the production of low scale high-quality products, but they are not suitable for big-scale production. For big-scale production, traditional manufacturing is suitable.

6.4. Engaging Patients with Rehabilitation Tools

We believe technology is a powerful tool to engage patients in the rehabilitation process. We observed that the bionic arm diverted attention from therapy exercises to observation and imitation of the bionic arm during our pilot study. We believe this is another advantage of modern rehabilitation aids in the rehabilitation process. Some researchers already report positive effects with the engagement of technology aids, for instance, older people with socially assistive technology in-home care [56]. Engaging patients [57] in rehabilitation with technology (techniques, tools, exercise sequence, etc.) is a fascinating research field that needs further investigation.

6.5. Share Your Ideas with the Community

Before producing a rehabilitation hand, a 3D model needs to be designed, for which 3D Computer-Aided Design (CAD) software is needed. There are plenty to choose from, from the commercial to the free ones. Free and open-sourced is FreeCAD. Fusion360 is a simple tool to start with, and is free for non-commercial projects. We used that tool (see

Figure 11) to design our rehabilitation hand. Before printing, the 3D model was converted in a format that the 3D printer understands (usually in gcode).

This design can be shared with the community. If someone has the same needs as we do, he can use our design to 3D print the rehabilitation hand. The rehabilitation platform already supports that hand. No further actions are needed in the communication layer of RehabHand platform. If we decide for a different design, we must be careful with the functionality currently supported with the rehabilitation platform. The limitation here is that the rehabilitation hand has the same functionality as the one already included in the platform. However, sharing these designs with the community will extend the range of supported rehabilitation devices, and most likely, one could find a design serving his/her needs.

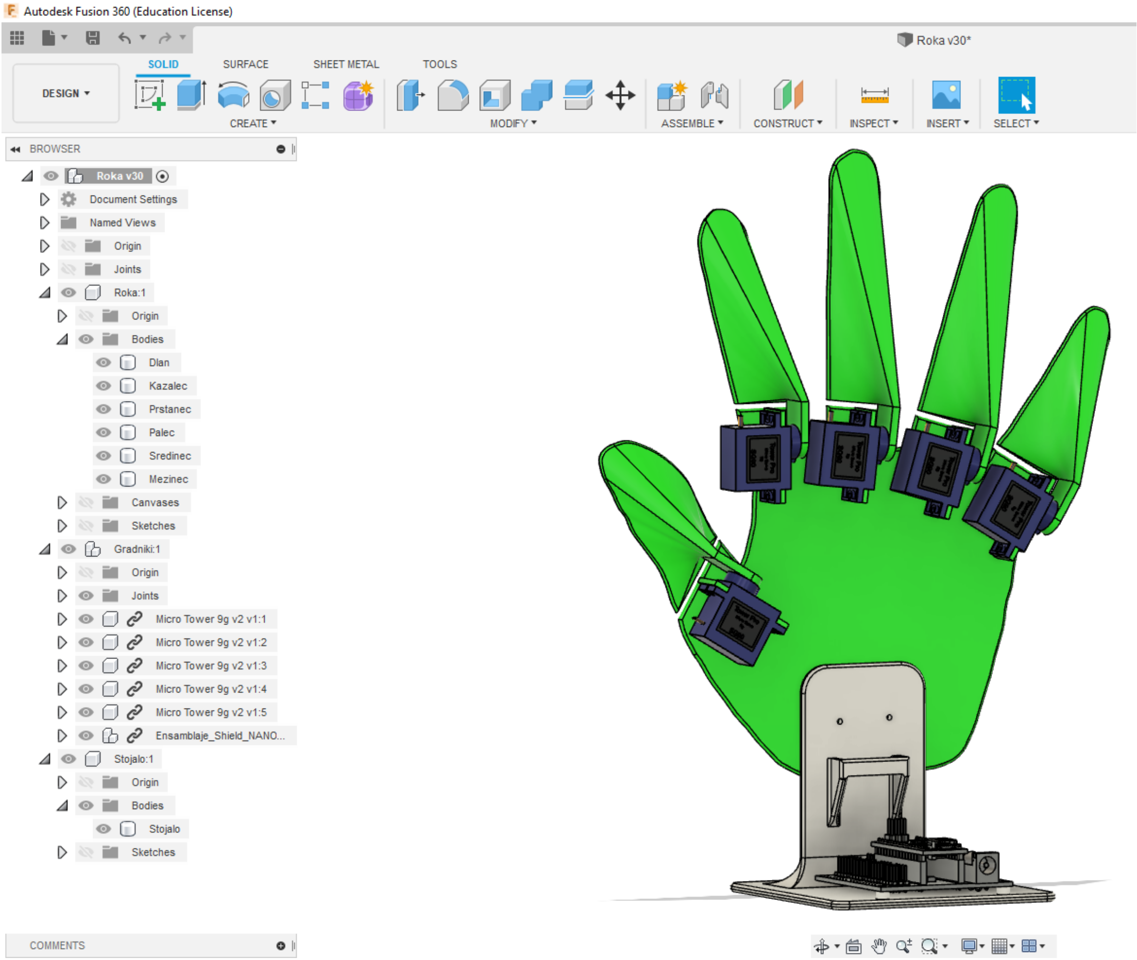


Figure 11. Rehabilitation hand design in Fusion360.

7. Related Work

This Section presents recent efforts done in rehabilitation aids and compares them with RehabHand from different viewpoints.

7.1. Summary of Related Work in Rehabilitation Aids

In Reference [58], Virtual Reality (VR) technology is presented for use in the hand rehabilitation process. The author presents a game developed in Unity controlled with a hand gesture detection device (LEAP motion). Reference [58], similarly to RehabHand uses a 3D printed bionic hand. This hand is used as a simulator for people who will need to use bionic limbs. To observe movements, the rehabilitation hand is manufactured using 2 cameras and 3 infrared LEDs that follow hand, wrist, and elbow position, one of our *must support* future work activities. The paper presents a game-based rehabilitation aid that helps train and strengthens the hand. Our work differs from Reference [58] since it attempts to provide a platform where different devices can be plugins, not only a VR solution.

Reference [35] advocates video games and devices like Kinect as useful rehabilitation tools. Similarly, RehabHand uses observation and repetition of movements, illustrated with a rehabilitation hand. The 3D depth-sensing camera of Kinect provides a possibility to develop games with balance training exercises. We included Kinect and similar video games devices in our rehabilitation aid platform as one of the possible simulations.

7.2. Additive Manufacturing in Rehabilitation

The authors of Reference [50] highlight patients' co-designing of assistive technology using 3D printing. They argue that commercially available devices do not always match a specific patient's needs, and sometimes it is easier to develop customized assistive technology. They hypothesize that digital fabrication empowers patients with physical limits to become active in personal assistive technology provision for their own needs. We use the idea of additive manufacturing and apply it to IT technology-supported rehabilitation aids. Additionally, we envision a complete cycle of different roles, including therapists and software developers, to develop future rehabilitation aids that can be used at home.

7.3. Summary of Related Work on DSLs

To the best of our acknowledge, we do not know any work where DSL is applied to the field of Rehabilitation Aids. However, DSLs are used in many other areas like Sport [59], Civil Engineering [60], Astronomy [61], Biology [62], Medicine [63], and so forth. In the following subsection, some papers are discussed that share commonalities with the RehabHand platform.

Synthetic biology is research that seeks to create new biological parts, devices, and systems, or to redesign systems found in nature. In Reference [62] the authors present a Domain-Specific Language called Eugene, meant to design synthetic biological designs (based on biological parts, devices, and rules). The benefit from a formalized and rigorous specification is to use it (together with other tools) to design space exploration, simulation, and automated assembly in this area. Eugene and RehabHand both share readability for humans. Both are intended for end-user development (Eugene for a Synthetic Biologist, RehabHand for Therapists). Both DSLs are also the executable specification.

VR-MED is a Domain-Specific Language that aims to develop serious games applied to Family Medicine teaching [63]. VR-MED is defined with a formalism for medical learning specification, and is used to derive games from these cases. Similar to RehabHand, VR-MED is intended for non-programming users. Contrary to RehabHand, games are constructed using graphical language elements, while RehabHand uses textual notation.

End-user development is also supported in Reference [64], where the solution RoboGen enables medication management using a Healthcare Robot. Similarly to RehabHand, a solution provides a close-loop involving different users with the following roles [64]—Physician (generate prescription), caregiver, older person, and pharmacist. A RoboGen has a defined workflow, and enables end-user development for the following services: Main medication instructions by the doctor, medication names by the doctor and verification by the caregiver, medication timings and schedule by doctor and caregiver, side effects monitoring questions by the doctor, patient educational information by the clinicians, doctor appointments and refill reminders by a caregiver.

8. Conclusions

Studying literature, we found many interesting tools in rehabilitation [3,35,50,56]. However, none of them support different hardware alternatives, connecting hardware of diverse manufacturers. Our platform is the first attempt to support the community and provide a platform to have a uniform software that is loosely coupled with the underlying software and hardware. We live in an era where advances in electronics, robotics, IT, and additive manufacturing are enormous. With our platform, we enable the independent layers, so the hardware can be switched, or even supported, by additive manufacturing, which, in our opinion, might change the future of health care. The other novelty is an independent rehabilitation exercise definition, which was shown in the example of hand rehabilitation. This enables easier, customized exercise definition for an individual user, not dependent on the software front-end.

We know there are limitations in the current work (number of supported rehabilitation devices, etc.). Therefore, we envision many interesting research directions. Regarding code generation, we want to investigate how language composition techniques [65] can assist us in the code generation process for different types of rehabilitation devices. To test our rehabilitation platform, we intend to extend it with assistive technology and robotic rehabilitation with hand exoskeleton technology. Another priority direction in our research is support for the evaluation of patient hand movements. We plan to extract neural information from the hand's surface, and use it for further rehabilitation processes [66] and define customized exercises for the individual. We already have some results from a preliminary study with a hand glove. However, we are still seeking alternatives and studying motion assessment obtained from hand flexion and extension, joints and tissue stretch to use this information back in the rehabilitation process. On the other hand, RehabHand needs to be evaluated from different viewpoints. We want to check whether engaging with a rehabilitation aid and robot-assisted rehabilitation result in increased engagement, confidence, and motivation levels of patients in a rehabilitation process. In the future, we would like to spread our technology to other physical disabilities. For instance, we would like to test children with cerebral palsy [25] and other neuromotor disabilities in more frequent and intensive therapy by doing personalized exercises with rehabilitation aids.

Author Contributions: Conceptualization, T.K., M.M. and M.Č.; Investigation, T.K., M.M., M.H. and M.Č.; Software, T.K. and M.H.; Writing—original draft, T.K.; Writing—review & editing, Z.L., M.M., M.H. and M.Č. All authors have read and agreed to the published version of the manuscript.

Funding: The Slovenian Research Agency supported this work under Grants P2-0041 and BI-CN/18-20-007. This work was also supported by the Changshu Science and Technology Development Programme under Grant CR201719.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: RehabHand source code is available at: https://github.com/tomazkosar/RehabHand. Additional information is available from the corresponding author upon reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. van de Sandt-Koenderman, W.M.E. Aphasia rehabilitation and the role of computer technology: Can we keep up with modern times? *Int. J. Speech Lang. Pathol.* **2011**, *13*, 21–27. [CrossRef] [PubMed]
2. Stanica, I.C.; Moldoveanu, F.; Portelli, G.P.; Dascalu, M.I.; Moldoveanu, A.; Ristea, M.G. Flexible Virtual Reality System for Neurorehabilitation and Quality of Life Improvement. *Sensors* **2020**, *20*, 6045. [CrossRef] [PubMed]
3. Marques, B.; McIntosh, J.; Valera, A.; Gaddam, A. Innovative and Assistive eHealth Technologies for Smart Therapeutic and Rehabilitation Outdoor Spaces for the Elderly Demographic. *Multimodal Technol. Interact.* **2020**, *4*, 76. [CrossRef]
4. Hwang, J.; Kim, K.H.; Hwang, J.G.; Jun, S.; Yu, J.; Lee, C. Technological Opportunity Analysis: Assistive Technology for Blind and Visually Impaired People. *Sustainability* **2020**, *12*, 8689. [CrossRef]
5. Wang, Q.; Chen, W.; Markopoulos, P. Literature review on wearable systems in upper extremity rehabilitation. In Proceedings of the IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI), Valencia, Spain, 1–4 June 2014; pp. 551–555.
6. Burdea, G. Virtual rehabilitation-benefits and challenges. *Yearb. Med. Inform.* **2003**, *12*, 170–176.
7. Yakub, F.; Khudzari, A.Z.M.; Mori, Y. Recent trends for practical rehabilitation robotics, current challenges and the future. *Int. J. Rehabil. Res.* **2014**, *37*, 9–21. [CrossRef] [PubMed]

8. Kietzmann, J.; Pitt, L.; Berthon, P. Disruptions, decisions, and destinations: Enter the age of 3-D printing and additive manufacturing. *Bus. Horiz.* **2015**, *58*, 209–215. [CrossRef]
9. Nyberg, E.L.; Farris, A.L.; Hung, B.P.; Dias, M.; Garcia, J.R.; Dorafshar, A.H.; Grayson, W.L. 3D-printing technologies for craniofacial rehabilitation, reconstruction, and regeneration. *Ann. Biomed. Eng.* **2017**, *45*, 45–57. [CrossRef]
10. Lin, Y.C.; Huang, L.Y.; Chen, C.S. Strength Evaluation and Modification of a 3D Printed Anterior Ankle Foot Orthoses. *Appl. Sci.* **2020**, *10*, 7289. [CrossRef]
11. Lunsford, C.; Grindle, G.; Salatin, B.; Dicianno, B.E. Innovations with 3-dimensional printing in physical medicine and rehabilitation: A review of the literature. *PM&R* **2016**, *8*, 1201–1212.
12. Burgar, C.G.; Lum, P.S.; Shor, P.C.; Van der Loos, H.M. Development of robots for rehabilitation therapy: The Palo Alto VA/Stanford experience. *J. Rehabil. Res. Dev.* **2000**, *37*, 663–674. [PubMed]
13. Maciejasz, P.; Eschweiler, J.; Gerlach-Hahn, K.; Jansen-Troy, A.; Leonhardt, S. A survey on robotic devices for upper limb rehabilitation. *J. Neuroeng. Rehabil.* **2014**, *11*, 3. [CrossRef] [PubMed]
14. Balasubramanian, S.; Klein, J.; Burdet, E. Robot-assisted rehabilitation of hand function. *Curr. Opin. Neurol.* **2010**, *23*, 661–670. [CrossRef] [PubMed]
15. Krebs, H.I.; Ladenheim, B.; Hippolyte, C.; Monterroso, L.; Mast, J. Robot-assisted task-specific training in cerebral palsy. *Dev. Med. Child Neurol.* **2009**, *51*, 140–145. [CrossRef] [PubMed]
16. Meng, W.; Liu, Q.; Zhou, Z.; Ai, Q.; Sheng, B.; Xie, S.S. Recent development of mechanisms and control strategies for robot-assisted lower limb rehabilitation. *Mechatronics* **2015**, *31*, 132–145. [CrossRef]
17. Mernik, M.; Heering, J.; Sloane, A. When and How to Develop Domain-Specific Languages. *ACM Comput. Surv.* **2005**, *37*, 316–344. [CrossRef]
18. Kosar, T.; Mernik, M.; Carver, J. Program comprehension of domain-specific and general-purpose languages: Comparison using a family of experiments. *Empir. Softw. Eng.* **2012**, *17*, 276–304. [CrossRef]
19. Kosar, T.; Bohra, S.; Mernik, M. Domain-Specific Languages: A Systematic Mapping Study. *Inf. Softw. Technol.* **2016**, *71*, 77–91. [CrossRef]
20. Kosar, T.; Bohra, S.; Mernik, M. A Systematic Mapping Study driven by the margin of error. *J. Syst. Softw.* **2018**, *144*, 439–449. [CrossRef]
21. Fischer, G.; Giaccardi, E.; Ye, Y.; Sutcliffe, A.G.; Mehandjiev, N. Meta-design: A manifesto for end-user development. *Commun. ACM* **2004**, *47*, 33–37. [CrossRef]
22. Fowler, M. *Domain-Specific Languages*; Pearson Education: New York, NY, USA, 2010.
23. Kelly, S.; Tolvanen, J.P. *Domain-Specific Modeling: Enabling Full Code Generation*; John Wiley & Sons Inc.: Hoboken, NJ, USA, 2008.
24. O'Sullivan, S.B.; Schmitz, T.J.; Fulk, G. *Physical Rehabilitation*; FA Davis: Philadelphia, PA, USA, 2019.
25. Lu, Z.; Ma, Z.; Borovac, B.; Mernik, M. Design of cerebral palsy rehabilitation training system based on speech recognition technology to control robotic hand movements. *Chin. High Technol. Lett.* **2019**, *29*, 503–511.
26. Jesus, T.S.; Landry, M.D.; Dussault, G.; Fronteira, I. Human resources for health (and rehabilitation): Six rehab-workforce challenges for the century. *Hum. Resour. Health* **2017**, *15*, 8. [CrossRef] [PubMed]
27. Bernd, T.; Van Der Pijl, D.; De Witte, L. Existing models and instruments for the selection of assistive technology in rehabilitation practice. *Scand. J. Occup. Ther.* **2009**, *16*, 146–158. [CrossRef] [PubMed]
28. Preece, J.; Sharp, H.; Rogers, Y. *Interaction Design: Beyond Human-Computer Interaction*; John Wiley & Sons: Hoboken, NJ, USA, 2015.
29. Lu, Z.; Jiang, R.; Ma, Z.; Borovac, B.; Mernik, M. Face-expression and speech recognition based rehabilitation training system. *Chin. High Technol. Lett.* **2019**, *29*, 287–294.
30. Lieberman, H.; Paternò, F.; Klann, M.; Wulf, V. End-user development: An emerging paradigm. In *End User Development*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 1–8.
31. Svrcek, M.; Kompan, M.; Bielikova, M. Towards understandable personalized recommendations: Hybrid explanations. *Comput. Sci. Inf. Syst.* **2019**, *16*, 179–203. [CrossRef]
32. Khalajzadeh, H.; Simmons, A.J.; Abdelrazek, M.; Grundy, J.; Hosking, J.; He, Q. An end-to-end model-based approach to support big data analytics development. *J. Comput. Lang.* **2020**, *58*, 100964. [CrossRef]
33. Kajan, E.; Faci, N.; Maamar, Z.; Sellami, M.; Ugljanin, E.; Kheddouci, H.; Stojanović, D.H.; Benslimane, D. Real-time tracking and mining of users' actions over social media. *Comput. Sci. Inf. Syst.* **2020**, *17*, 403–426. [CrossRef]
34. Lu, Z.; Hong, W.; Ma, Z.; Shen, X.; Penčić, M.; Yan, L.; Shan, C.; Long, G.; Bin, L.; Mernik, M. Design of digital training system for cerebral palsy rehabilitation based on speech recognition. *Chin. High Technol. Lett.* **2020**, *30*, 526–532.
35. Lange, B.; Koenig, S.; McConnell, E.; Chang, C.Y.; Juang, R.; Suma, E.; Bolas, M.; Rizzo, A. Interactive game-based rehabilitation using the Microsoft Kinect. In Proceedings of the Virtual Reality Conference, Costa Mesa, CA, USA, 4–8 March 2012; pp. 171–172.
36. Gershenfeld, N. How to make almost anything: The digital fabrication revolution. *For. Affairs* **2012**, *91*, 43.
37. Xu, X.; Duguleana, M. Trajectory Planning of 7-Degree-of-Freedom Manipulator Based on ROS. *IOP Conf. Ser. Mater. Sci. Eng.* **2019**, *677*, 052072. [CrossRef]
38. van Deursen, A.; Klint, P.; Visser, J. Domain-Specific Languages: An Annotated Bibliography. *ACM SIGPLAN Not.* **2000**, *35*, 26–36. [CrossRef]

39. Mernik, M. An object-oriented approach to language compositions for software language engineering. *J. Syst. Softw.* **2013**, *86*, 2451–2464. [CrossRef]
40. de la Vega, A.; García-Saiz, D.; Zorrilla, M.; Sánchez, P. Lavoisier: A DSL for increasing the level of abstraction of data selection and formatting in data mining. *J. Comput. Lang.* **2020**, *60*, 100987. [CrossRef]
41. Bentley, J. Little Languages. *Commun. ACM* **1986**, *29*, 711–721. [CrossRef]
42. Kosar, T.; Oliveira, N.; Mernik, M.; Varanda Pereira, M.J.; Črepinšek, M.; da Cruz, D.; Henriques, P.R. Comparing General-Purpose and Domain-Specific languages: An Empirical Study. *Comput. Sci. Inf. Syst.* **2010**, *7*, 247–264. [CrossRef]
43. Kosar, T.; Martínez López, P.E.; Barrientos, P.A.; Mernik, M. A preliminary study on various implementation approaches of domain-specific language. *Inf. Softw. Technol.* **2008**, *50*, 390–405. [CrossRef]
44. Hudak, P. Modular Domain Specific Languages and Tools. In Proceedings of the Fifth International Conference on Software Reuse, Victoria, BC, Canada, 5 June 1998; pp. 134–142.
45. Parr, T. *The Definitive ANTLR 4 Reference*; Pragmatic Bookshelf: Raleigh, NC, USA, 2013.
46. Levine, J.R.; Mason, T.; Brown, D. *Lex & Yacc*; O'Reilly Media, Inc.: Newton, MA, USA, 1992.
47. Mernik, M.; Lenič, M.; Avdičaušević, E.; Žumer, V. LISA: An Interactive Environment for Programming Language Development. In *Compiler Construction*; Horspool, R.N., Ed.; Springer: Berlin/Heidelberg, Germany, 2002; pp. 1–4.
48. Porubän, J.; Forgáč, M.; Sabo, M. Annotation based parser generator. In Proceedings of the 2009 International Multiconference on Computer Science and Information Technology, Mragowo, Poland, 12–14 October 2009; pp. 707–714.
49. Bettini, L. *Implementing Domain-Specific Languages with Xtext and Xtend*; Packt Publishing Ltd.: Birmingham, UK, 2016.
50. Thorsen, R.; Bortot, F.; Caracciolo, A. From patient to maker—A case study of co-designing an assistive device using 3D printing. *Assist. Technol.* **2019**, 1–7. [CrossRef]
51. De Couvreur, L.; Dejonghe, W.; Detand, J.; Goossens, R. The role of subjective well-being in co-designing open-design assistive devices. *Int. J. Des.* **2013**, *7*, 57–70.
52. Hofmann, M.; Burke, J.; Pearlman, J.; Fiedler, G.; Hess, A.; Schull, J.; Hudson, S.E.; Mankoff, J. Clinical and maker perspectives on the design of assistive technology with rapid prototyping technologies. In Proceedings of the 18th International ACM Sigaccess Conference on Computers and Accessibility, Reno, NV, USA, 23–26 October 2016; pp. 251–256.
53. Watanabe, T.; Hatakeyama, T.; Tomiita, M. Improving Assistive Technology Service by Using 3D Printing: Three Case Studies. *Stud. Health Technol. Inform.* **2015**, *217*, 1047–1052.
54. Wang, T.; Zhang, D.; Da, L. Remote-controlled vascular interventional surgery robot. *Int. J. Med. Robot. Comput. Assist. Surg.* **2010**, *6*, 194–201. [CrossRef]
55. World Health Organization. *Global Spending on Health: A World in Transition*; Technical Report; World Health Organization: Geneva, Switzerland, 2019.
56. Khosla, R.; Chu, M.T.; Khaksar, S.M.S.; Nguyen, K.; Nishida, T. Engagement and experience of older people with socially assistive robots in home care. *Assist. Technol.* **2019**, 1–15. [CrossRef] [PubMed]
57. Lamrani, R.; Abdelwahed, E.H. Game-based learning and Gamification to improve skills in early years education. *Comput. Sci. Inf. Syst.* **2019**, *17*, 339–356. [CrossRef]
58. Alimanova, M.; Borambayeva, S.; Kozhamzharova, D.; Kurmangaiyeva, N.; Ospanova, D.; Tyulepberdinova, G.; Gaziz, G.; Kassenkhan, A. Gamification of hand rehabilitation process using virtual reality tools: Using leap motion for hand rehabilitation. In Proceedings of the 2017 First IEEE International Conference on Robotic Computing (IRC), Taichung, Taiwan, 10–12 April 2017; pp. 336–339.
59. Fister, I., Jr.; Fister, I.; Mernik, M.; Brest, J. Design and implementation of domain-specific language easytime. *Comput. Lang. Syst. Struct.* **2011**, *37*, 151–167. [CrossRef]
60. Lee, J.K.; Eastman, C.M.; Lee, Y.C. Implementation of a BIM domain-specific language for the building environment rule and analysis. *J. Intell. Robot. Syst.* **2015**, *79*, 507–522. [CrossRef]
61. Lesher, K.M. A Domain-Specific Language for Documentation of Amateur Astronomy Observations. Ph.D. Thesis, Kutztown University of Pennsylvania, Kutztown, PA, USA, 2019.
62. Bilitchenko, L.; Liu, A.; Cheung, S.; Weeding, E.; Xia, B.; Leguia, M.; Anderson, J.C.; Densmore, D. Eugene—A domain specific language for specifying and constraining synthetic biological parts, devices, and systems. *PLoS ONE* **2011**, *6*, e18882. [CrossRef]
63. Mossmann, J.B.; Rieder, R.; Flores, C.D.; Pinho, M. Project and Preliminary Evaluation of VR-MED, a Domain-Specific Language for serious games in Family Medicine Teaching. In Proceedings of the 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), Atlanta, GA, USA, 10–14 June 2016; Volume 2, pp. 663–667.
64. Datta, C.; Yang, H.Y.; Tiwari, P.; Kuo, I.H.; MacDonald, B.A. End User Programming to Enable Closed-Loop Medication Management Using a Healthcare Robot. *Soc. Sci.* **2011**. Available online: https://www.researchgate.net/profile/Chandan_Datta2/publication/262917013_End_User_Programming_to_Enable_Closed-loop_Medication_Management_Using_a_Healthcare_Robot/links/0046353945e0c8cae3000000.pdf (accessed on 1 January 2021).
65. Erdweg, S.; Giarrusso, P.G.; Rendel, T. Language Composition Untangled. In Proceedings of the Twelfth Workshop on Language Descriptions, Tools, and Applications (LDTA'12), Tallinn, Estonia, 31 March–1 April 2012; p. 7.
66. Akdogan, E.; Shima, K.; Kataoka, H.; Hasegawa, M.; Otsuka, A.; Tsuji, T. The cybernetic rehabilitation aid: Preliminary results for wrist and elbow motions in healthy subjects. *IEEE Trans. Neural Syst. Rehabil. Eng.* **2012**, *20*, 697–707. [CrossRef]