VITHEA Kids 2.0

An application to help children with communication disorders

Cláudia Patricia Balixa Filipe

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur
Prof. José Alberto Rodrigues Pereira Sardinha

Examination Committee

Chairperson: Prof. Nuno João Neves Mamede
Supervisor: Prof. Maria Luísa Torres Ribeiro Marques da Silva Coheur
Member of the Committee: Prof. Alberto Abad Gareta

June 2017
Acknowledgments

I would like to thank my parents, for their dream of making their daughter go to college and be whatever she dreamed, no matter how hard it was, for keeping me in the line and never letting me give up. I would also like to thank my grandmother for her unconditional support and love.

I must thank to my dissertation supervisors Prof. Maria Luisa Torres Ribeiro Marques da Silva Coheur and Prof. José Alberto Rodrigues Pereira Sardinha, for challenging me, believe in me, fight with me and get here with me.

It was also not possible to be here without Vânia Mendonça, who was tireless during the whole process, or without Filipe Sardinha, who is always there for me.

Last but not least, to all my friends and colleagues who believe in better, and make work and study possible together, thank you.

To each and every one of you – Thank you.
Abstract

Communication disorders include deficits in language, speech and communication. These deficits can also be caused by other related disorders such as Autism Spectrum Disorder (ASD). Individuals with communication disorders face numerous adversities in their daily life, which can be minimized through solving exercises. Some efforts have been made to develop applications that can provide these exercises and also be suitable for its users, in particular for children with ASD.

An application developed in this context is the Virtual Therapist for Autism Treatment on Children (VITHEA Kids), which is a European Portuguese application for helping children with ASD and their caregivers providing the possibility of creating exercises and customizing various aspects of interaction with the platform, as well as a talking animated character.

In order to continuously grow, as technological tool, VITHEA Kids required a software reengineering to continue providing the best user and developer experience.

This master thesis project has three main contributions to VITHEA Kids: 1) a software reengineering, which supported 2) making improvements to existent functionalities and 3) adding new functionalities related with: a) prompting, since children will need some support on learning new skills, b) reinforcement, motivating children to practice the targeted behavior, and c) emotions, due to the impact of emotions on learning. Also, d) a set of exercises, with functional thematics was included in the application (e.g., family, daily tasks).

Keywords

Communication disorders; Autism Spectrum Disorder; Communication skills; Software reengineering; Computer-assisted learning.
Resumo

As perturbações de comunicação incluem dificuldades na linguagem, discurso e comunicação. Estas dificuldades podem estar relacionadas com outras perturbações, como uma Perturbação do Espectro do Autismo. Indivíduos com perturbações de comunicação enfrentam diversas adversidades no seu dia-a-dia, o que pode ser minimizado com a resolução de exercícios. Diversos esforços têm sido feitos para desenvolver aplicações que possam providenciar exercícios e que sejam também adequados para os utilizadores, em particular para crianças com autismo.

Uma aplicação desenvolvida neste contexto é o VITHEA Kids, uma aplicação em Português Europeu, que tem como objetivo ajudar as crianças com autismo e os seus cuidadores, proporcionando a possibilidade de criar exercícios e personalizar vários aspectos de interação com a aplicação, assim como uma personagem animada que fala. Para continuar a crescer, sendo uma ferramenta tecnológica, o VITHEA Kids precisa de passar por um processo de reengenharia do seu software, para continuar a proporcionar a melhor experiência tanto para o utilizador como para o programador.

O projeto desta tese de mestrado é composto por três principais contribuições para o VITHEA Kids: 1) um processo de reengenharia de software, o que permitiu 2) melhorar as funcionalidades existentes e 3) adicionar novas funcionalidades, relacionadas com: a) ajuda, para apoiar as crianças enquanto adquirem novas habilidades, b) reforço, para motivar as crianças a praticar o comportamento pretendido, e c) emoções, dado o impacto que as emoções têm na aprendizagem. Adicionalmente, d) foi adicionado um conjunto de exercícios com temáticas funcionais (por exemplo, família, dia-a-dia, tarefas diárias).

Palavras Chave

Perturbações de comunicação; Perturbação do Espectro do Autismo; Habilidades de comunicação; Reengenharia de software; Aprendizagem assistida por computador.
Contents

1 Introduction ................................................................. 1
   1.1 Motivation .......................................................... 3
   1.2 Goals ................................................................. 4
   1.3 Document structure ............................................... 5

2 Background ............................................................... 7
   2.1 Communication Disorders ......................................... 9
   2.2 ASD - Autism Spectrum Disorder .................................. 10
   2.3 VITHEA Kids - Virtual Therapist for Autism Treatment on Children ........................................... 12
      2.3.1 Caregiver’s module ............................................. 14
      2.3.2 Child’s module ................................................ 15
      2.3.3 Points to improve .............................................. 16

3 Related Work ............................................................ 19
   3.1 Software reengineering ........................................... 21
   3.2 Applied Behaviour Analysis ....................................... 24
      3.2.1 Prompting ..................................................... 24
      3.2.2 Reinforcement ............................................... 25
   3.3 Emotions ............................................................. 27

4 VITHEA Kids 2.0 ......................................................... 31
   4.1 Reengineering VITHEA Kids ..................................... 33
      4.1.1 Security ....................................................... 36
   4.2 Improvements ........................................................ 36
   4.3 New functionalities ................................................ 37
      4.3.1 Prompting module ............................................ 37
      4.3.2 Reinforcement module ...................................... 38
      4.3.3 Emotions’ module ............................................ 38
5 Evaluation

5.1 Developer experience ................................................. 43
5.1.1 In the previous architecture ..................................... 43
5.1.2 In the new architecture ........................................... 47
5.2 Improvements ......................................................... 53
5.3 New functionalities .................................................. 56
5.3.1 Prompting module ................................................. 56
5.3.2 Reinforcement module ............................................ 57
5.3.3 Emotions’ module ................................................ 57
5.3.4 Exercises set ...................................................... 58
5.4 User testing ........................................................... 58
5.4.1 Quality assurance tests .......................................... 58
5.4.2 Heuristic evaluation ............................................. 59
5.4.3 Pilot testing ....................................................... 59
5.4.4 Usability testing .................................................. 60

6 Conclusion ............................................................... 63

6.1 Future work ............................................................ 65
List of Figures

2.1 VITHEA Kids - Caregiver’s module .................................................. 12
2.2 VITHEA Kids - Child’s module .......................................................... 13
3.1 Comparing Play Framework with Spring, Ruby on Rails and Node.js .......... 23
3.2 iPrompts® ......................................................................................... 25
3.3 iReward ............................................................................................. 27
3.4 Ekman’s basic emotions facial expressions plus neutral [1] ....................... 29
3.5 ABA Flash Cards & Games - Emotions ............................................... 30
3.6 Emotionary by Me.Mu ...................................................................... 30
4.1 Play Framework error reporting .......................................................... 34
5.4 Caregiver’s personal area .................................................................. 53
5.6 Exercise forms .................................................................................. 54
5.7 Pagination, search and filter ............................................................. 55
List of Tables

3.1 Spring vs. Play Framework ............................................. 22
3.2 Fundamental emotions’ theories (adapted from [2]) .................. 29
### Listings

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Routes example</td>
<td>22</td>
</tr>
<tr>
<td>5.1</td>
<td>Create table CaregiverChild</td>
<td>43</td>
</tr>
<tr>
<td>5.2</td>
<td>Caregiver and Child models</td>
<td>44</td>
</tr>
<tr>
<td>5.3</td>
<td>Spring configuration file - part I</td>
<td>45</td>
</tr>
<tr>
<td>5.4</td>
<td>ManageCaregiverAction</td>
<td>46</td>
</tr>
<tr>
<td>5.5</td>
<td>Spring configuration file - part II</td>
<td>46</td>
</tr>
<tr>
<td>5.6</td>
<td>JSP page</td>
<td>47</td>
</tr>
<tr>
<td>5.7</td>
<td>Struts configuration file</td>
<td>47</td>
</tr>
<tr>
<td>5.8</td>
<td>Entity bean - Child</td>
<td>48</td>
</tr>
<tr>
<td>5.9</td>
<td>Entity bean - Caregiver</td>
<td>49</td>
</tr>
<tr>
<td>5.10</td>
<td>AdminChilCtrl controller</td>
<td>50</td>
</tr>
<tr>
<td>5.11</td>
<td>Child client model</td>
<td>50</td>
</tr>
<tr>
<td>5.12</td>
<td>Children component</td>
<td>51</td>
</tr>
</tbody>
</table>
## Acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABA</td>
<td>Applied Behaviour Analysis</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ASD</td>
<td>Autism Spectrum Disorder</td>
</tr>
<tr>
<td>BO</td>
<td>Business Object</td>
</tr>
<tr>
<td>CAL</td>
<td>Computer-aided learning</td>
</tr>
<tr>
<td>CLI</td>
<td>Command-line Interface</td>
</tr>
<tr>
<td>CRUD</td>
<td>Create, Read, Update and Delete</td>
</tr>
<tr>
<td>CSRF</td>
<td>Cross-Site Request Forgery</td>
</tr>
<tr>
<td>CSS</td>
<td>Cascading Style Sheets</td>
</tr>
<tr>
<td>DAO</td>
<td>Data Access Object</td>
</tr>
<tr>
<td>DR</td>
<td>Differential reinforcement</td>
</tr>
<tr>
<td>DSM</td>
<td>Diagnostic and Statistical Manual of Mental Disorders</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>HTTPS</td>
<td>Hypertext Transfer Protocol Secure</td>
</tr>
<tr>
<td>IDE</td>
<td>Integrated Development Environment</td>
</tr>
<tr>
<td>JPA</td>
<td>Java Persistence API</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>L2F</td>
<td>Spoken Language Systems Laboratory</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>MVC</td>
<td>Model View Controller</td>
</tr>
<tr>
<td>NIDCD</td>
<td>National Institute on Deafness and Other Communication Disorders</td>
</tr>
<tr>
<td>ORM</td>
<td>Object-relational mapping</td>
</tr>
<tr>
<td>PROPOR</td>
<td>International Conference on the Computational Processing of Portuguese</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>SPA</td>
<td>Single Page Application</td>
</tr>
<tr>
<td>VITHEA Kids</td>
<td>Virtual Therapist for Autism Treatment on Children</td>
</tr>
<tr>
<td>VITHEA</td>
<td>Virtual Therapist for Aphasia Treatment</td>
</tr>
</tbody>
</table>
1 Introduction

Contents

1.1 Motivation .............................................. 3
1.2 Goals .................................................... 4
1.3 Document structure ................................. 5
This thesis presents a solution to improve and add new functionalities to an existing application for children with Autism Spectrum Disorder (ASD) and their caregivers: the Virtual Therapist for Autism Treatment on Children (VITHEA Kids). The proposed solution intends to be helpful for children diagnosed with Autism Spectrum Disorder (ASD), which requires the application to be designed according to all the specificities of this disorder, but also for neurotypical children\(^1\) (i.e., children who do not have any developmental disabilities).

The present section describes the motivation for this thesis, the goals intended to be achieved and the structure of the present document.

### 1.1 Motivation

Communication allows human beings to express their feelings and needs, likes and dislikes, to interact and build relationships, thus being an extremely important part of a human development. Individuals with communication disorders face numerous adversities in their daily lives. According to the Diagnostic and Statistical Manual of Mental Disorders (DSM) [3], “Disorders of communication include deficits in language, speech, and communication”. The data compiled by National Institute on Deafness and Other Communication Disorders (NIDCD) shows that 7.5 million people in the US have some trouble using their voices, and 8 to 9% of young children have some speech sound disorder, like stuttering. Also, between 6 and 8 million people in the US have some form of language impairment.

In Portugal, according to an epidemiological study [4] (with 748 children between five and eleven years old), 48,2% of the children presented some speech or language alteration, of which:

- 34,1% are articulation alterations;
- 12,2% are language alterations;
- 1,1% is voice alterations;
- 0,8% is resonance, breathing and stuttering alterations, and not conclusive assessment.

In its fifth edition, DSM differentiates communication disorders from other disorders that also induce difficulties to communication, like ASD, which, due to its particular characteristics, will be the main disorder addressed in this project.

Some of the existent difficulties can be minimized through solving exercises that help to develop some skills (e.g., word naming, generalization skills). Therefore, several therapy exercises have been developed, intending to minimize the inherent difficulties of communication. Most of these therapies

\(^1\)https://en.wikipedia.org/wiki/Neurotypical, last access 01/01/2016
are based in a psychology approach called Applied Behaviour Analysis (ABA), which uses interventions based in concepts like prompting and reinforcement, combined with others.

Also, given the particular interest of children with ASD on computers and software [5, 6], several Computer-aided learning (CAL)\(^2\) applications have been developed, first for computers, then for tablets and smartphones. The developers' goal is to promote the interventions' practice, with reduced monetary costs, and make the applications easy to use and carry.

Despite the growing popularity of CAL applications, there are still gaps that can be filled, as it has been mentioned by Mendonça [7], namely:

- Most applications are paid or include paid features;
- Few applications offer these contents in European Portuguese;
- User's progress is not commonly taken into account;
- There are few customization options.

To fulfil these gaps, Mendonça developed VITHEA Kids [7, 8], which is an application for helping children with ASD to improve their language and communication skills. The main goal of this project is to continue that effort, contributing to VITHEA Kids’ growth, as described in Section 1.2.

### 1.2 Goals

The main goal of this work is to provide an application that comprises enough exercises and custom options to make it suitable for its users (children with ASD) and caregivers, such as professionals and therapists or even parents or families that will value a helpful tool. Also, this tool should be free and easy to use and take with.

VITHEA Kids is a promising system to achieve this main goal, since it is in European Portuguese, free, customizable and designed taking into account some feedback received from caregivers and individuals with these difficulties.

This project's goals are:

- Reengineering VITHEA Kids’ software since, as technological tool, it should keep providing the best user’s and developer’s experience;
- Improving VITHEA Kids usability and functionality through the fulfilment of several gaps that the system presented, regarding both the system’s architecture and the user interface design;

\(^2\)https://en.wikipedia.org/wiki/Computer-aided, last access 06/05/2017
• Adding new functionalities to VITHEA Kids:
  
  – **Prompting module:** prompting is a technique used in the traditional exercises (i.e., exercises that do not resort to software applications), that consists of helping the child (e.g., providing cues). The prompting module is aimed at customizing the prompts that will be used in the exercises of VITHEA Kids;
  
  – **Reinforcement module:** reinforcement is also used in traditional exercises, typically by rewarding the child. This module will provide options about the rewards and the rewarding strategies;
  
  – **Emotions’ module:** children with ASD face emotions’ related difficulties, and emotions can also be useful in learning. This module will include emotions in VITHEA Kids, through an existing animated character and by providing new exercises about emotions;

• Creating a set of exercises, with different thematics related with functionality;

• Evaluating the reengineering process regarding the developer's experience, by describing what changed in the new application's version, and prepare a user testing session to assess the acceptance of the new modules, the impact of the changes and the users' satisfaction when interacting with VITHEA Kids.

### 1.3 Document structure

Presented the motivation and main goals of this project, the remainder of the document will describe the background concepts (in the Background section), which will allow to fully understand the scope of the work and VITHEA Kids, the system that will support this project. In Related Work section, a reengineering process is exemplified, and the most relevant subjects that will support the new modules will be addressed, along with other applications that already use those concepts. In VITHEA Kids 2.0 section, the proposal is presented with the full description both conceptual and architectural. Finally, in Evaluation section, the results are presented and compared, and a user testing session is proposed. The document ends with the Conclusion section, presenting some conclusions, highlights and future work.
## Background

### Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1 Communication Disorders</td>
<td>9</td>
</tr>
<tr>
<td>2.2 ASD - Autism Spectrum Disorder</td>
<td>10</td>
</tr>
<tr>
<td>2.3 VITHEA Kids - Virtual Therapist for Autism Treatment on Children</td>
<td>12</td>
</tr>
</tbody>
</table>
To fully understand the context around this project, the present section will provide background concepts, like the main ones about Communication Disorders and ASD. In a second part, there will be a description of VITHEA Kids as it will be the baseline for this project.

2.1 Communication Disorders

A communication disorder is an impairment in the ability to receive, send, process, and comprehend concepts or verbal, nonverbal and graphic symbol systems [9] and it can be manifested as a deficit in speech, language and/or hearing.

Speech is the expressive production of sounds, including the articulation, fluency, voice and resonance quality [3]. An articulation disorder is related to the production of sounds and may result in substitutions, omissions, additions or distortions that can make the speech less intelligible\(^1\).

On the other hand, a fluency disorder consists in the interruption in the flow of speaking, like atypical rate, rhythm and repetitions in sounds, syllables, words or phrases.

The disorder could also be related to the voice itself, being considered a voice disorder, and it is characterized by the low vocal quality, pitch, loudness resonance and/or duration inappropriate for and individual's age and/or gender.

To illustrate speech disorders, it is worth mentioning:

- **Speech Sound Disorders**: difficulties in coordinating movements of the articulators (jaw, tongue and lips) with breathing and voice, and/or phonological difficulties. It is typically diagnosed if a 4 years-old child's speech is not intelligible and there is no other condition associated.

- **Childhood-Onset Fluency Disorder (Stuttering)**: frequent disturbances in the normal fluency and time patterning of speech, like sound and syllable repetitions, sound prolongations and broken words (e.g., “I-I-I-I see him”) [10]. It is usually more severe in special pressure situations for the individual.

The way an individual communicates in terms of form, function and symbols (like spoken/written words, signs for sign language, pictures) composes what it is referred to as Language [3]. A Language Disorder is not only about the production of the symbols, but also about its comprehension. According to DSM [3], the impairment may be related with:

- **Form**, which, in spoken language, is about phonology and, in written language, is about morphology, i.e., the rules that govern sound combinations and/or words' structure and construction

---

\(^1\)Intelligibility is a measure of how comprehensible speech is in given conditions.
respectively, and the syntax, which is the order, combination and relation of words in a sentence.

- **Content of language**, meanings of words and sentences, i.e., semantics.

- **Function of language**, using language to functional and/or social purposes.

An example of disorder related to impairments in Language’s function is **Social (Pragmatic) Communication Disorder**, which consists in persistent difficulties in the social use of communication [3], for example, greeting, sharing information, adapting to different places (classroom vs. playground), respecting conversation rules, and so on. In a general way, language disorder manifests through difficulties in acquisition and use of spoken, written, signing or another language, showing a lack of vocabulary and sentence structure, and discourse impairments.

**Hearing** is a physiological capability to perceive sound. Impairments at the auditory system level may limit the development, comprehension, production and/or maintenance of speech and/or language. Hearing disorders can be related to detection, recognition, comprehension and perception of the sound at different levels, describing individuals as **Deaf** or **Hard of hearing**, either the individual relies on the auditory channel as primary sensory input for communication or not.

### 2.2 ASD - Autism Spectrum Disorder

Besides Communication Disorders, there are other disorders associated with communication difficulties, such as **ASD**, which is a spectrum for a set of developmental disabilities that can cause significant social, communication, and behavioral challenges. So, children diagnosed with it may have difficulties developing language skills and understanding what is said to them [11].

Eugen Bleuler was the first person to use the **autism** term, which comes from a Greek word *autos*, meaning “self”, around 1911 to refer to one group of symptoms of schizophrenia. Leo Kanner described Autism in 1943 [12], exemplifying the case of eleven children (eight boys and three girls) who had common behaviors among them, but not common with children of their age. Autism was then described as the “**inability to relate themselves in an ordinary way to people and situations from the beginning of life**” considering that those children were in an “**autistic aloneness**”, ignoring the external stimulus or reacting as if they cause them pain. Likewise, there is a need for being left undisturbed and everything that is brought from the outside, or changes the environment, represents a dreaded intrusion.

Regarding communication, eight of the eleven children started to speak, either at the normal age or after some delay. The other three did not learn to speak at all. However, none of the talking children
used it to communicate with others. All, except one, were capable of clear articulation, phonation\(^2\) and name objects, learn words, rhymes, prayers, lists of animals, the alphabet or even a foreign language, but it took a long time to put words together. When finally sentences were formed, they were for a long time repetitions of heard word combinations (which can be heard and echoed immediately or uttered later - delayed echolalia). Also, learning concepts took many years, since these children associate a concept to a particular and literal situation (e.g., one of the children learned that his father would put him in the shoulders if he said “yes”, after father asked if he want to, so for him “yes” only has this meaning). Likewise, personal pronouns are repeated just as heard (e.g., when a child wants some milk or dessert, he utters “Now I will give you your milk” and “Are you ready for dessert?” because once his parents uttered that way). This phenomenon happened with all the eight speaking children. Despite the fact that these children could echo things they had heard before, they do not easily attend to what is said to them. It requires numerous reiterations of the question or command before there is even so much as an echoed response (seven of the children were considered as deaf or hard of hearing).

In 1944, Hans Asperger also described a group of children with the same behaviors as Kanner. However, he also mentioned those children had an exceptionally good development in what concerns to cognitive and language skills [13,14].

In the following decades there were some theories about autism’s causes, namely the theory of “refrigerator mothers”, which relates autism with the emotional distance between the children and their mothers, and only in the 1970’s autism was considered a separate syndrome, distinguishing it from intellectual disability, schizophrenia and from other developmental disorders.

Nowadays, ASD is described in DSM as having two principal characteristics that limit everyday functioning and are present from early childhood:

- Persistent deficits in social communication and interaction, namely, deficits in social-emotional (e.g., the kid, in Kanner’s, article, who would not react when his parents held him in their arms), in nonverbal communication (e.g., facial expressions and eye contact), and the absence of interest in peers.

- Restricted, repetitive patterns of behavior, interests, or activities, which can be illustrated, among others, as:
  - repetitive motor movements, lining toys or objects, repeating words;
  - keeping the sameness: the routines, eat the same food, take some route;

\(^2\)Sound production through vocal folds.
– high focus capability and interest in details.

This definition covers all the previous ones, e.g., Kanner’s autism and Asperger’s disorder, into the autistic spectrum.

2.3 VITHEA Kids - Virtual Therapist for Autism Treatment on Children

VITHEA Kids is an application for helping children with ASD to improve their language and communication skills, and their caregivers to assist them with it, providing a tool that allows creating a set of exercises that can be of many types (e.g., image identification, matching an image with expression/word, finding the outsider), according to the skill that is supposed to be practice (vocabulary acquisition, word-picture association or generalization). These exercises are created by the caregiver on the Caregiver’s module and can be solved by the child on the Child’s module.

![VITHEA Kids - Caregiver’s module](image1)

The Caregiver’s module allows managing exercises, images, users information, both children and caregivers. There are also options to customize the interaction between the caregiver and the child. When creating an exercise, the caregiver can attribute a topic to it (e.g., “Animals”), a difficulty level (e.g., Introductory, Intermediate or Advanced), the question/instruction, the stimulus, the correct answer and a set of possible answers (distractors) (Figure 2.1a). Also, the caregiver can list and edit the existent exercises, as well as the list of child users with some relevant information, like name, birth date and gender (Figure 2.1b).

VITHEA Kids has also an animated character, Catarina, and her utterances (e.g., greeting or congratulating) can be set by the caregiver for each child. Another customizable option is the set of rein-
forcement images to be used when the child answers correctly to a question.

On the Child’s module, the child (or caregiver) can select the type of exercise he/she wants to “play” (Figure 2.2a), and the first exercise will be presented by Catarina, which utters the question, and shows an optional complementary image (stimulus), and a set of possible answers (text), where one of them is correct and the others are distractors (Figure 2.2b). If the child selects the correct answer on the first try, the reinforcement image will show up else the distractor disappears to help the child to select the correct answer and the correct answer is highlighted, along with Catarina uttering the remaining options (Figure 2.2c).

(a) Choose exercise’s type  
(b) Exercise  
(c) Prompting

Figure 2.2: VITHEA Kids - Child’s module

In terms of specifications, VITHEA Kids has a client-server architecture and comprises both desktop (web) and mobile (Android) applications. The web application is deployed with Apache Tomcat, follows a Model View Controller (MVC) layer architecture, and has a MySQL database where the information is stored.

It is worth mentioning that this architecture was based on another system, Virtual Therapist for Aphasia Treatment (VITHEA)\(^3\)\(^4\), which is another application related to communication skills’ related disorders. VITHEA [15, 16] was created to help individuals with Aphasia\(^5\), which is a set of communication disorders caused by brain damages or strokes, and it is diagnosed in individuals that had normal communication skills, namely language skills, before. In this case, speech exercises, in general, could help slowing or receding the disorder’s evolution. The exercises structure consist in presenting an image,

\(^3\)VITHEA is an award-winner platform developed by Spoken Language Systems Laboratory (L2F)
\(^4\)https://vithea.l2f.inesc-id.pt, last access 09/12/2015
\(^5\)https://en.wikipedia.org/wiki/Aphasia, last access 09/12/2015
text, video or audio to the patient (stimulus), and ask him a question about it (e.g., to name the object presented in the stimulus). The answer is provided orally by the patient and it is considered correct if it matches with one of the multiple correct answers defined by the therapist, or one of its synonyms or diminutives, even if it contains unrelated elements (by using a technique called Keyword spotting\(^6\)). Farther, the exercise can be of many different categories, e.g., to associate a caption with an image, to complete a sentence or to relate a part as a whole, according to the skill that is supposed to be practiced.

The following sections will describe in detail the caregiver’s module architecture (Section 2.3.1, the child’s module architecture (Section 2.3.2), and the identified problems and users’ requests as points to improve (Section 2.3.3).

### 2.3.1 Caregiver’s module

The caregiver’s module is a Web application, deployed with Apache Tomcat (a Java Web Server), which persists data in a MySQL database. The Tomcat’s web container handles the requests from servlets\(^7\), in this case, the Java Server Pages (JSP files which embed Java code in Hypertext Markup Language (HTML) files). This architecture is illustrated in Figure 2.3.

![Figure 2.3: A first-look architecture](image)

Thinking about a system’s architecture, it should be arranged in layers, separating different concerns\(^8\) (e.g., the data layer should be separated from the “business logic” and presentation layers). The architecture described in Figure 2.3 can induce mixing the presentation layer with the “business logic” layer\(^9\), and so it is not recommended for high scale projects. Thus, VITHEA and VITHEA Kids developers separated the model (data layer), view (presentation layer) and controller (“business logic” layer) using Apache Struts 2\(^10\). Using Struts, requests are sent to a controller (actions), which calls the respective Action class that interacts with the model. The model returns an Action Forward for the controller to

---

\(^6\)https://en.wikipedia.org/wiki/Keyword_spotting, last access 01/01/2016  
\(^7\)https://en.wikipedia.org/wiki/Java_servlet, last access 27/06/2016  
\(^8\)https://en.wikipedia.org/wiki/Software_architecture, last access 12/08/2016  
\(^9\)https://en.wikipedia.org/wiki/Apache_Struts_1, last access 27/06/2016  
\(^10\)https://struts.apache.org/, last access 27/06/2016
know which tile\textsuperscript{11} it should send to the client. A tile is a part of the web page (presentation layer) that is loaded without having to refresh the whole page. The caregiver’s module also uses Spring\textsuperscript{12}, a framework that intends to make Java programming easier, safe and efficient, using several design patterns\textsuperscript{13}, and Hibernate\textsuperscript{14}, another framework that helps to map the database schema with the Model classes. Full architecture is represented in Figure 2.4.

![Figure 2.4: VITHEA Kids architecture](image)

### 2.3.2 Child’s module

The child’s module is an Android application, which makes Hypertext Transfer Protocol (HTTP) requests to an Application Programming Interface (API). The user interface of the application has two main parts side-by-side: the Catarina’s view (the animated character) and the exercises view, where the menus and exercises are displayed. Technically speaking, the application is composed of two Android projects. The main project imports Catarina’s project (which is a Unity scene\textsuperscript{15}) into the main activity\textsuperscript{16}, which will occupy the half of the view, and the sequences and exercises representation will be represented in the remaining activity’s half. Each time the application needs to request something to the server, an AsyncTask\textsuperscript{17} is created, which will be responsible for making the request, while a loading dialog is presented to the user.

\textsuperscript{11}Tiles: template mechanism to build the views (supported by Struts), http://tiles.apache.org/, last access 27/06/2016
\textsuperscript{12}https://spring.io/, last access 27/06/2016
\textsuperscript{13}https://www.quora.com/Why-we-should-use-Spring, last access 26/07/2016
\textsuperscript{14}http://hibernate.org/, last access 27/06/2016
\textsuperscript{15}https://docs.unity3d.com/Manual/CreatingScenes.html, last access 20/04/2017
\textsuperscript{16}https://developer.android.com/reference/android/app/Activity.html, last access 20/04/2017
\textsuperscript{17}https://developer.android.com/reference/android/os/AsyncTask.html, last access 30/04/2017
2.3.3 Points to improve

Although, the system’s architecture seems very robust and promising, in practical terms, the development was compromising by the usage of many frameworks. Some difficulties were early identified, during the first development tasks, namely:

- **Errors were not easy to understand**: when there is some error the application stops loading and, to find out the error it is necessary to search in a great dimension log file from Tomcat (catalina.out);

- **Solutions were not easy to find**: to find the solution for an error, or even the code where something was wrong or missing, was not trivial since there are many files and XML configuration files where the error could be and it is not possible to debug;

- **Performance was not the best**: after some page requests, the system turns very slow until it crashes and the page is not retrieved anymore. This also happens out of development environment, e.g. in tests environment causing a bad user experience;

- **Low developer’s productivity**: during the development, every change in the code requires a redeploy of the project so the changes take effect on the application. Also, Hibernate annotations were very hard to map with the database schema, causing many bugs and consuming much development time, and as an attempting to correct some of these bugs, it was necessary to adapt the database schema to map with the models and with the annotations, deviating from what a good database design should be;

- **Steep learning curve**: learn how all the frameworks work and co-work was hard, which may (or may not) contribute to the previous points, but it still is a valid point since VITHEA Kids is a project with a great potential that should be easy to add new modules and functionalities without much of an effort.

In this context, with the continuous application improvement and growing goals in mind and considering some new technologies and frameworks, we consider that a software reengineering is required for VITHEA Kids.

Regarding user experience, VITHEA Kids has some limitations in both caregiver’s and child’s modules:

- **Caregiver’s sharing records**: All the data in the system is public or shared between all caregivers, i.e., if a caregiver edits an exercise, the changes will be reflected in the instance of another caregiver, as well as the progress of the children, which can be accessed by all the caregivers.
• **Missing some Create, Read, Update and Delete (CRUD) operations:** Not all the operations are implemented or need to be reviewed for exercises, children and multimedia resources.

• **Few customization options:** The application lacks customization options, which are extremely important given the nature of children with ASD.

• **Image exercises integration:** This development is not fully integrated with the child’s application.

• **Records’ lists are too long:** The listing pages (e.g., exercises list) should use pagination instead of present to the caregiver a long list with a long scroll.

• **Missing forms useful tools:** The creation and edition forms (e.g., add an exercise) lack of visual hints (e.g., for mandatory fields) and helpful hints in case of error.

• **Not possible to preview exercises:** It should be possible to preview a created exercise.

• **Android versions not supported:** The child’s application crashed with earlier versions of Android (namely for Android 5.x\(^{18}\)).

• **Child’s application session is not kept:** It is not possible to keep the child’s session between usages, which means that the child has always to log in.

Also, there are functionalities requested by the caregivers who try VITHEA Kids:

• **Arrange exercises into sequences:** It should be possible to organize exercises (by an order) into sequences (e.g., lesson one and lesson two).

• **Navigate through exercises:** In the child’s application, it should be possible to skip the current exercise and return to the previous one.

• **Customize animated character:** Regarding the animated character it would be pleasant to change the background image, according to the child’s preferences, or even better, to be able to use Filipe or Edgar (from ChatWoz \(^{17}\)).

\(^{18}\)[https://www.android.com/history](https://www.android.com/history), last access 02/01/2017

\(^{17}\)[17]
3

Related Work

Contents

3.1 Software reengineering .......................................................... 21
3.2 Applied Behaviour Analysis ...................................................... 24
3.3 Emotions ............................................................................. 27
The present section will address the most relevant subjects which based the software reengineering, improvements and new functionalities proposed to VITHEA Kids, remembering that the main target of this system are children with ASD. Regarding the reengineering process, the case of LinkedIn\textsuperscript{1} software reengineering will be presented. Also, as the most of the therapies developed are based in ABA, which is a psychology approach that uses interventions based on prompting and reinforcement, these concepts will be analyzed, as well as its impact in exercises. Regarding the meaning (or lack of it) that emotions have to children with ASD, the impact of introducing emotions in educational applications, like VITHEA Kids will be a study object.

3.1 Software reengineering

Currently, there are many open source\textsuperscript{2} frameworks available, which by being open source they have a large on-line community, i.e., people that write blog articles, make video talks and create tutorials, for those frameworks, which is useful for the developer since it helps to solve problems and make decisions. Yevgeniy Brikman is a software engineer that worked at LinkedIn and wrote some blog articles in 2013 about how LinkedIn changed its platform from Spring to Play Framework\textsuperscript{3}, a web framework based on a lightweight, stateless, web-friendly architecture, that intends to make it easier to build web applications in Java and Scala. Brikman wrote an article pointing out some reasons for this decision\textsuperscript{4} by comparing Spring and Play Framework (Table 3.1), where the problems identified in Spring and solved by Play Framework are the same difficulties identified and described in Section 2.3.3.

Play Framework seemed to solve the identified problems, but more frameworks had to be considered and analyzed. Brikman also did it and explained it at JaxConf 2013\textsuperscript{5,6}. He compared Play Framework with Spring, Ruby on Rails\textsuperscript{7} and Node.js\textsuperscript{8} in the following four scalability dimensions. The results are illustrated in Figure 3.1 (adapted from Brikman’s presentation\textsuperscript{4}).

- **Raw horsepower** (language and framework overhead): theoretical maximum performance for the server in ideal conditions;

- **Concurrent horsepower** (framework’s approach to concurrency): performance with many users, I/O and real world conditions;

\textsuperscript{1}https://en.wikipedia.org/wiki/LinkedIn, last access 06/05/2017
\textsuperscript{2}https://en.wikipedia.org/wiki/Open-source_software, last access 03/07/2016
\textsuperscript{3}https://engineering.linkedin.com/play/play-framework-linkedin, last access 27/06/2016
\textsuperscript{4}https://www.quora.com/Which-is-better-Play-Framework-or-Spring-MVC-How-should-I-decide-what-to-use, last access 27/06/2016
\textsuperscript{5}http://jaxconf.com/, last access 14/08/2016
\textsuperscript{6}https://www.youtube.com/watch?v=8z3h4Uv9YbE, last access 14/08/2016
\textsuperscript{7}http://rubyonrails.org/, last access 14/08/2016
\textsuperscript{8}https://nodejs.org/, last access 14/08/2016
Spring vs. Play Framework

<table>
<thead>
<tr>
<th>Developer productivity</th>
<th>Long time to start and to see a change (minutes)</th>
<th>Hot-reload changes (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Poor APIs</td>
<td>Powerful and composable APIs</td>
</tr>
<tr>
<td></td>
<td>Few built-in functions for building a modern application</td>
<td>Modern framework (built-in support for JSON, web sockets, DB access, web service calls, etc.)</td>
</tr>
<tr>
<td>I/O</td>
<td>Blocking (every request holds a thread)</td>
<td>Fully non-blocking</td>
</tr>
<tr>
<td>Functional programming</td>
<td>Many mutable objects, annotations, XML configuration files, etc., which makes it harder to identify errors or to custom settings</td>
<td>Everything is a function and returns a value: classes, routes, templates, build system, etc., which makes it easier to think, develop, reuse, debug and custom code</td>
</tr>
<tr>
<td>Error reporting</td>
<td>Big log file</td>
<td>Clear error message and information about where the error is</td>
</tr>
</tbody>
</table>

Table 3.1: Spring vs. Play Framework

- **Single developer** (framework’s raw productivity and tooling): how easy it is to get started and how quickly a single developer can build things;
- **Multiple developers** (code rot and maintainability): how the framework tolerates many developers working.

In terms of architecture, Play Framework is RESTful and follows MVC pattern. It allows to easily expose a Representational State Transfer (REST) API, i.e., a set of endpoints that map with the controller methods. The exert represented in Listing 3.1 illustrates the route file where this mapping is defined.

```java
GET /children controllers.ChildrenController.getChildren
POST /addChild controllers.ChildrenController.addChild
```

Listing 3.1: Routes example

The access to these endpoints is done using HTTP verbs (e.g., GET for getting data from server side, or POST to send and change data), which means that any layer that can make an HTTP call (and deal with the results, e.g. JavaScript Object Notation (JSON)) could use this API.

Regarding persistence, Play comes with Ebean ORM (an Object-relational mapping (ORM) tool), which allows generating the database schema based on model classes using Java Persistence API (JPA).

---

14 [http://ebean-orm.github.io/](http://ebean-orm.github.io/), last access 17/04/2017

annotations\textsuperscript{15,16}, just like Hibernate did in the previous architecture. Since Ebean is sessionless (i.e., does not require a persistence context to persist the data units, called entity beans) and it only uses a subset of JPA, it may be easier to use\textsuperscript{17,18,19}. Play also comes with Evolutions\textsuperscript{20}, which is a tool for tracking database changes and generate change scripts.

For LinkedIn, according to Brikman, Play was a good choice to balance performance, reliability and developer productivity.

\textsuperscript{15}\url{https://en.wikipedia.org/wiki/Java_Persistence_API}, last access 17/04/2017
\textsuperscript{16}\url{http://www.objectdb.com/api/java/jpa}, last access 17/04/2017
\textsuperscript{17}\url{http://ebean-orm.github.io/architecture/compare-jpa}, last access 17/04/2017
\textsuperscript{18}\url{http://ebean-orm.github.io/architecture/compare-hibernate}, last access 17/04/2017
\textsuperscript{19}\url{http://stackoverflow.com/a/20843625}, last access 17/04/2017
\textsuperscript{20}\url{https://www.playframework.com/documentation/2.5.x/Evolutions}, last access 17/04/2017
3.2 Applied Behaviour Analysis

ABA is a psychology approach that evaluates human behavior changes, regarding environmental stimuli\(^{21}\). This evaluation is done through interventions, which are based on operant conditioning principles [18], like prompting and reinforcement (which will change the behavior).

### 3.2.1 Prompting

**Prompting** is an antecedent intervention’s stimulus (precedes the behavior) that intends to help in order to improve the behavior. Prompting is important since it supports the child while developing the new behavior. A prompt can be an instruction, gesture, demonstration, touch or another stimulus. In a general way, prompts can be classified into [19]:

- **Verbal Prompts**: words, instructions or questions that approach the child to the correct answer/behavior. This type of prompt is the most common and can be used combined with other types;

- **Modeling**: demonstrations (models) of the behavior that is intended to be achieved. This is the second most common type and it is used combined with other types;

- **Manual Prompts**: physical guidance to help to perform the desired behavior. It required physical contact and most of the times is combined with other types too;

- **Gestural Prompts**: pointing to the correct answer or object, motioning (mimic), or nodding. This type of prompt is usually used to complement other main type of prompt.

- **Photographs and Line Drawings**: images, pictures and line drawings are used to teach behaviors like self-care and daily routines or time schedules. This type is usually combined with verbal prompts and models;

- **Textual Prompts**: can consist of checklists, scripts or written instructions. This type of prompt implies that the child already knows how to read;

- **Other types of prompt**: some investigators have already used tactile prompts, tones and alarms, and color cues too.

The prompting principle should be adapted to the circumstances since it is desirable that the child after some iterations does not need help anymore. The caregiver can start by always helping the child, and, after some iterations, only helping if the child does not give the correct answer after a while until it is not necessary to help (Most-to-Least strategy). There are other strategies like Least-to-Most.

\(^{21}\) **stimulus** (n., pl. -li) - something that incites or quickens action, feeling, thought, etc. (http://www.thefreedictionary.com/stimuli, last access 22/11/2015)
where the assistance is increased only if the child does not perform the desired behavior within a specific amount of time, **Delayed Prompts**, where the time interval between the stimulus and the prompt is increased until the child answers correctly before the prompt, **Graduate Guidance**, which uses manual prompts that are reduced by changing their intensity or location, **Stimulus Fading**, which uses exaggerated physical dimensions like color, size and intensity and fade them until they are no longer needed, and finally, **Stimulus Shaping**, where the physical characteristics of the stimulus are changed (e.g., for decimal numbers, first the representation may be “2 and 25”, and gradually the “and” is becoming smaller until it can be represented as “2,25” [19]).

Some companies developed applications to help children, based on the prompting principle, which use different types of prompts. For example, iPrompts®[22] is an application for iPhone and iPad, created for children with ASD and caregivers. This application allows caregivers to create and present visual content to help the children schedule their tasks, and transact between them providing visual schedules with pictures, text and audio, visual count down timer, choice prompts and modeling videos for school and social behavior, as illustrated in Figure 3.2.

![Schedules](image1)

![Count down timer](image2)

![Modeling videos](image3)

**Figure 3.2: iPrompts®**

### 3.2.2 Reinforcement

Contrary to prompting, **Reinforcement** is a consequent intervention’s stimulus (succeeds the behavior) that intends to strength a behavior (make it occur more frequently) providing a pleasant reinforcer (positive reinforcement) or removing an unpleasant one (negative reinforcement). A reinforced

behavior tends to be repeated (i.e., strengthened), but a behavior which is not reinforced tends to occur fewer times (i.e., weakened). The reinforcement can be delivery following different strategies (or schedules) [20]:

- **Continuous Reinforcement**: the reinforcement is always given when the target behavior occurs;
- **Fixed Ratio Reinforcement**: the reinforcement is given only after the target behavior occurs a specified number of times;
- **Variable Ratio Reinforcement**: the reinforcement is given after the target behavior occurs an unpredictable number of times;
- **Fixed Interval Reinforcement**: the reinforcement is given after a fixed time interval, during which the target behavior occurred;
- **Variable Interval Reinforcement**: the reinforcement is given after an unpredictable amount of time, during which the target behavior occurred;

The reinforcement principle is the basis for certain interventions [21]:

- **Token economy**: consists in an intervention where the rewards are given after a correct answer, called tokens, can be later traded for a real reward (e.g., a snack);
- **Extinction**: an intervention that intends to remove the habit of a certain behavior to be reinforced;
- **Differential reinforcement (DR)**: is an intervention that can be as simple as if the child answers correctly, the behavior is rewarded, otherwise, it is not. It can also be more complex like **DR of Higher Rates of Behavior** or **DR of Lower Rates of Behavior** where the reinforcer is given when the behavior occurs more or less often, respectively. Another strategy could be **DR of Other Behavior**, where the reward is given as long as an inappropriate behavior does not occur, **DR of Alternative Behavior** or **DR of Incompatible Behavior**, where the reward is given when a different behavior occurs. The difference between the latter two relies on the behavior: an alternative behavior is a better behavior than the usual one, opposed to an incompatible behavior that is better and cannot co-exist with the usual one (e.g., sitting instead of running).

Reinforcement is very common in most application types, especially in games and gamification[23]. A popular example would be the game Angry Birds[24] where the player can earn up to three stars on every level, but he/she only earns a star if he/she completes the goal of the level (destroy all pigs) and the number of stars earned is proportional to the performance of the player (i.e., the quality of the behavior).

---

[23] https://en.wikipedia.org/wiki/Gamification, last access 09/12/2015
The player needs to earn stars to unlock the next levels.

An example of an application that works on reinforcement is iReward\textsuperscript{25} that intends to reinforce positive behaviors using visual rewards. iReward allows the caregiver to define a behavior, e.g., “Get dressed yourself”, the number of times the child should do it, represented by stars, e.g., six stars, and the reward to earn, e.g., “Toy truck”, after the child performed the behavior the defined number of times (Figure 3.3a).

The child can see a list of behaviors/tasks, the respective rewards and progress (Figure 3.3b). The progress is also shown through the color of the stars, that are white in the beginning and become yellow once the child performs the behavior (Figure 3.3c).

![Figure 3.3: iReward](image)

Reinforcement, as it was presented (from behaviorist psychology), inspired the machine learning area to create Reinforcement Learning, where software agents drive their behavior in order to maximize the rewards. This new concept of reinforcement learning allowed to create computer models to investigate certain matters, for example, the role of emotion in learning \cite{22}.

### 3.3 Emotions

Since the first diagnostics of Autism, emotions were always a concern. Some children were labeled as emotionally disturbed and some emotion related theories were very popular (recap the theory of “refrigerator mothers” in Section 2.2). Even according to DSM \cite{10}, emotional impairments are part of the diagnosis as “\textit{marked impairments in the use of multiple nonverbal behaviors such as eye-to-eye gaze, facial expression, body postures, and gestures to regulate social interaction}”, “a

\textsuperscript{25}https://itunes.apple.com/us/app/ireward/id324643198, last access 29/11/2015
lack of spontaneous seeking to share enjoyment, interests, or achievements with other people (e.g., by a lack of showing, bringing or pointing out objects of interest)” and “a lack of social or emotional reciprocity”. The emotional impairments of children with ASD can be translated into different domains [23]:

- **Expressing emotions**: children with ASD have difficulty in describing what they are feeling, so they can even say that they do not feel a certain emotion, which contributes to a lack of facial expressions;

- **Perceiving and understanding emotions and facial expressions**: basic emotions (like happiness or sadness) are normally understood by children with ASD, in contrast to more subtle expressions like anger, fear, surprise or disgust, since children pay more attention to the mouth and less to the eyes, losing part of the expression information;

- **Responding to emotions**: children with ASD do not pay much attention to other people's emotions or expressions, and do not share points of interest, like an interesting object that the child would point out to another person or that the other person pointed out for the child (this is called lack of joint or shared attention).

In fact, emotions and face expressions can be complex concepts and not even theorists can achieve a consensus. Ortony & Turner [2] compiled some of those theories in a table (Table 3.2) which shows the basic or fundamental emotions identified by different theorists.

To identify these emotions, theorists used several techniques, including to find a distinctive facial expression related to each emotion. For example, Figure 3.4 illustrates Ekman [1] six basic emotions plus the neutral emotion.

Emotions can be classified as positive or negative if they are pleasant or unpleasant, and as activating or deactivating, if they are beneficial or devastating. Positive emotions like enjoyment, excitement, hope and pride have an impact on learning, since they affect children's attention and motivation, and encourages the use of learning strategies and self-regulation of learning [24]. For example, the pride of a good grade will turn the child's attention to that goal, i.e., if the emotion is task-related, like enjoyment or excitement of learning, the child will be focused on that task. Again, emotions are complex since a positive emotion can help focus on the task or drive the child's attention away. For motivation and interest, activating positive emotions, like enjoyment, is very helpful in recollecting positive memories that help to remember the positive value of the task and the child's competence to solve it, in contrast with relaxation and relief, which are deactivating positive emotions that can reduce motivation. This is also true for the use of learning strategies and self-regulation of learning since activating positive

---


27 [https://en.wikipedia.org/wiki/Joint_attention](https://en.wikipedia.org/wiki/Joint_attention), last access 06/12/2015
emotions helps to employ flexible, creative and deep learning strategies (e.g., elaboration of learning material) and enhance flexible thought and action, like planning and monitoring learning activities, promoting self-regulation, in contrast to deactivating positive emotions which can reduce any systematic
use of learning strategies and inhibit self-regulation behaviors.

On the other hand, both activating negative emotions, like anxiety, anger and shame, and deactivating negative emotions, like hopelessness and boredom, contributes to decrease children’s attention. However, activating negative emotions can sometimes help to increase motivation, as long as the child is expecting to succeed, to facilitate the use of rigid learning strategies (like rehearsal and rote memorization) and to look for external regulation of learning, since negative emotions reduce child’s flexible thought and action. Deactivating negative emotions reduces motivation and any use of strategies or self-regulation.

There are many applications for learning emotions²⁸, for example, ABA Flash Cards & Games - Emotions²⁹ which is based on ABA therapy, using flash cards to identify emotions (Figure 3.5).

![Figure 3.5: ABA Flash Cards & Games - Emotions](http://www.smartappsforkids.com/2015/03/top-15-free-apps-for-learning-emotions.html, last access 07/12/2015)

A slightly different application is Emotionary³⁰, where children can register their emotions and try to relate them to some event (e.g., “I’m feeling happy because I ate an icecream”, Figure 3.6), allowing the child to recognize and understand those emotions.

![Figure 3.6: Emotionary by Me.Mu](https://itunes.apple.com/us/app/emotionary-by-me.mu/id555381720, last access 07/12/2015)
4

VITHEA Kids 2.0

Contents

4.1 Reengineering VITHEA Kids ................................................. 33
4.2 Improvements ................................................................. 36
4.3 New functionalities ......................................................... 37
VITHEA Kids was created for helping children with ASD to develop communication skills, however, there is a need for a software reengineering, and there are some improvements that can be done and some functionalities that can be added to the application. Regarding this, the present section describes this project intentions to contribute to VITHEA Kids.

4.1 Reengineering VITHEA Kids

Section 3.1 described how Play Framework solved LinkedIn identified problems regarding development. Expecting that Play Framework could solve VITHEA Kids’ development problems too, an analysis using the same points from Section 3.1 was made, regarding VITHEA Kids:

- **Developer’s productivity:** As identified in Section 2.3.3, in VITHEA Kids using Spring, every change requires a redeploy to reproduce the changes on the platform. Using Play, the changes can be seen in seconds through the hot-reload system (when the code is changed, the code is compiled and the web page refreshes automatically).

- **I/O:** Spring is blocking, which means that any time there is a request, it holds up a thread blocking the I/O, making the system crash after some requests. So changing for Play will improve the performance since it is a non-blocking framework making calls to remote services.

- **Functional programming:** Another mentioned difficulty was related to all the files and configuration files that needed attention when adding a new functionality. With Play, everything is a function that returns a value which makes it easier to develop and find problems in the code.

- **Error reporting:** Another pointed difficulty was identifying errors. With Play, errors are presented in browser and also in Command-line Interface (CLI) for more details (Figure 4.1), instead of the great dimension log file.

Play seems a promising framework for VITHEA Kids. Although, regarding the application’s front-end, and despite Play having an HTML engine, it can fall short as the user interface is enriched and consequently more complex. There are many frameworks that could help building a reactive web application, which consists of building a fast web application that ensures a positive user experience (not crashing, being slow or showing inconsistent data).

Angular is a JavaScript framework maintained by Google and a vast community. In terms of development, Angular makes it easier for programmers helping to create appealing interfaces with few

---

4. https://goo.gl/y9hQUE, last access 15/08/2016
5. https://angular.io/, last access 05/07/2016
code lines, also passing some of the back-end’s burden to the front-end, since there is no need for the server side to generate the HTML. For users, the experience improves, creating what is called a Single Page Application, which is faster since the resources are loaded only once (HTML, JavaScript and Cascading Style Sheets for style) reducing the number of requests to the server (and bandwidth usage). As mentioned, there are many JavaScript frameworks or libraries that could help achieve a rich interface. One of the most frequent comparisons is between Angular and React, despite React being a library and Angular a framework, which means that React can be seen as a collection of functionality that the developer controls, while Angular has a predefined flow that uses developer’s code (this is called Inversion of Control). Based on that, the choice relies on Angular for the front-end.

---

8 https://pt.wikipedia.org/wiki/Cascading_Style_Sheets, last access 13/09/2016
9 https://facebook.github.io/react/, last access 13/09/2016
10 http://martinfowler.com/bliki/InversionOfControl.html, last access 13/09/2016
framework to be used on VITHEA Kids. Even after choosing Angular, it was necessary to choose between AngularJS\(^\text{11}\) and Angular2, though with the growing popularity of the second, it seemed to be the most reasonable choice\(^\text{12}\), being one the main reasons the TypeScript\(^\text{13, 14}\), the programming language used. TypeScript is a JavaScript superset backed by Microsoft\(^\text{15}\), which improves developer performance since it adds compile-time type checks, object-oriented programming patterns (classes and interfaces) and typing features to JavaScript\(^\text{16}\). Also, it is supported by most of the Integrated Development Environment (IDE)\(^\text{17}\) which improves programmer experience (e.g., the IDE provides auto-complete hints and identifies some errors).

In conclusion, using Play as back-end framework will allow re-using some of the existing Java code (for back-end), putting it inside of a "black box" with much fewer configuration files, becoming easier to develop. On the other hand, using Angular2 as front-end framework will enrich the front-end side and improve the user experience. Figure 4.2 shows the proposed architecture.

![Proposed architecture](image)

Figure 4.2: Proposed architecture

This modular architecture is also useful for the child module since the current presentation layer can be replaced by a mobile application that makes the same HTTP requests and knows how to parse the JSON answers. Thus, the biggest change compared to the previous architecture, regarding the child's application, is to share the base code with the caregiver's application, since both applications consume the API (rather than have two different applications, which meant two different projects with the same logic to maintain).

\(^{11}\)https://angularjs.org/, last access 14/09/2016
\(^{12}\)https://dzone.com/articles/typed-front-end-with-angular-2, last access 14/09/2016
\(^{13}\)https://www.typescriptlang.org/, last access 15/09/2016
\(^{15}\)https://www.microsoft.com/, last access 15/09/2016
\(^{17}\)https://github.com/Microsoft/TypeScript/wiki/TypeScript-Editor-Support, last access 15/09/2016
4.1.1 Security

Exposing an API makes possible to anyone to access its methods through HTTP requests, so it is necessary to secure those endpoints. An architecture where Single Page Application (SPA) consuming a REST API allows improving security comparing to traditional applications, as pointed by James Ward in an article about “Securing Single Page Apps and REST Services”\(^{18}\). Instead of relying on a cookie to identify the logged in caregiver, which may be vulnerable to Cross-Site Request Forgery (CSRF) attacks, that token will be sent on every HTTP request header over an encrypted connection (i.e., using HTTPS).

In every request the authentication is verified, meaning that only authenticated caregivers can make requests. When the caregiver logs in, the API validates the credentials and returns an authentication token. The authentication token is stored in the browser. For each request made, the API identifies the caregiver by the token sent in the header, matching with the authentication token associated with the caregiver’s login data, stored and encrypted in the database.

4.2 Improvements

Given the points identified in the Section 2.3.3, the proposed improvements are:

- **Caregiver’s personal area**: It is necessary to implement a login system which will allow the caregiver to have a personal area, where he can have his exercises, children records and multimedia resources.

- **CRUD operations**: These operations, for exercises, children and multimedia resources records, will be reviewed and implemented when missing.

- **Preferences**: As fitting the application for each child is important, a menu with preferences will be available with options related to the animated character and the new modules.

- **Image exercises**: Multiple-choice exercises with image answers will be fully integrated with the child’s application.

- **Pagination**: The children, exercises and multimedia resources lists will have pagination (instead of showing too many on the same page).

- **Forms**: There will be visual hints for mandatory fields and help indications in case of error in the forms’ fields.

- **Preview exercises**: The caregiver will be able to preview the created exercises.

\(^{18}\)http://www.jamesward.com/2013/05/13/securing-single-page-apps-and-rest-services, last access 18/04/2017
• **Android version**: Since the application is not able to run in more than 50% of Android users\(^{19}\), it was necessary to address this issue. The crash is related to Catarina’s scene and the Unity’s project needs to be rebuilt, with a different target version. After that, it will be possible to run the child’s application in the most recent Android versions.

• **Child’s application session**: This issue is causing a poor user experience, since the session is not kept between the application usages, meaning that the user had to login every time he wanted to use the application. After implementing the new authentication, it will be possible to store the token in shared preferences\(^{20}\) and the session will be kept between usages. The logout method will clean the token from the shared preferences.

• **Sequences**: It will be possible for the caregiver to create sequences of exercises and sort the exercises inside the sequence.

• **Navigate through exercises**: It will be possible for the child to skip the current exercise and return to the previous exercise, for every exercise inside the current sequence.

• **Animated character**: The caregiver will be able to choose which animated character will be present in the child’s application: Catarina, Edgar and Filipe.

These changes will require deep changes in both caregiver’s and child’s application, from the data storage (database schema) to the back-end and front-end sides.

### 4.3 New functionalities

The current section describes the newly proposed functionalities, which will be added to VITHEA Kids, enhancing the application.

#### 4.3.1 Prompting module

The system helps the child making disappear wrong answers when the child fails, highlighting the right answer when the caregiver turns that option on and uttering all the possible answers through Catarina. Given the benefits of prompting, this proposal intends to add new options about it, like the type of prompting to be used and the strategy to be followed (remember Section 3.2.1), so the caregiver can define different options for each child he/she takes care of. The following types of prompting will be available:

• Change the visual look of the possible answers, either in size or color. Examples:

\(^{19}\)https://developer.android.com/about/dashboards/, last access 02/04/2017

\(^{20}\)https://developer.android.com/training/basics/data-storage/shared-preferences.html, last access 03/05/2017
– The correct answer (text or image) is bigger than the others;
– The correct answer (image) is in color, while the distractors are in black and white;
– The correct answer (text) has a color that pleases more the child (defined by the caregiver).

• Mark with a cross the wrong answers;
• The distractors disappear;
• Catarina reads the remaining possible answers;

Also, the caregiver will be able to choose the prompting strategy he/she wants to use, adapted from the known prompting strategies:

• **Always**: the prompting is always given;
• **If needed**: the prompting is given when the child selects a wrong answer;
• **No prompting**.

The types of prompting can be combined (e.g., the correct answer is the image in colors and with the great size). The strategies are mutually exclusive except the Stimulus Fading, which can be combined with one of the others.

### 4.3.2 Reinforcement module

In the reinforcement module, the caregiver should have the option to choose the strategy for applying the reinforcers (remember Section 3.2.2) for each child. The available strategies adapted from the existing ones, will be:

• **Always**: the reinforcer is always given, even if the child does not choose the correct answer at the first attempt;
• **At first**: the reinforcer is given only when the child chooses the correct answer at the first attempt;
• **No reinforcement**.

### 4.3.3 Emotions’ module

As emotions are helpful for learning and given the difficulties that children with **ASD** face regarding them, it is part of this proposal to address this topic in **VITHEA Kids**. Catarina already supports some facial expressions characteristic of the following emotions: neutral, joy, sadness and surprise, which can be helpful to provide feedback to the child as reinforcement after an exercise.
Besides that, a set of exercises about emotions will be added to help children identifying emotions and facial expressions. These exercises follow the currently supported structure of exercises, i.e., an image representing an emotion for the child to select the correct caption, or many images and a question to select what image answers to that question. Although the application has the support for these exercises, the resources to create them are hard to find, once we are talking about people’s facial expressions and for using the images in VITHEA Kids would be necessary to ensure that every image is under Creative Commons\textsuperscript{21} rights, which may be hard to manage. Having this in mind, we have proposed and managed an activity for the Portfolio course, which consisted in collect facial expressions images in the campus community (students and family). This activity was performed by the students Joana Condeço, Ana Salta and Nuno Baptista. In addition to collecting the images, students had to make sure that each individual previously agreed with making his photos available for the research community by signing a form where it was clear. This activity allowed to create a public database with facial expressions available on-line\textsuperscript{22}. It is worth mentioning that, during the activity it was possible to conclude that, even for neurotypical individuals it is hard to, given an emotion, produce the correspondent facial expression, or after collect the image with the facial expression, to identify the emotion. The collect images will be used in the set of exercises.

\textsuperscript{21}https://en.wikipedia.org/wiki/Creative_Commons, last access 07/05/2017
\textsuperscript{22}https://www.flickr.com/photos/vk_emotions/, last access 07/05/2017
5 Evaluation

Contents

5.1 Developer experience ....................................................... 43
5.2 Improvements ................................................................. 53
5.3 New functionalities .......................................................... 56
5.4 User testing ................................................................. 58
This section goal is to present the results of the changes proposed in Section 4. First, in order to compare the development in the old and new architecture, an exercise of developing the same functionality in both architectures will be presented, allowing to compare the developer’s experience. Then, the result of the implemented improvements and new functionalities will be provided for comparison with the previous application version. The section will be concluded with a proposal for the evaluation with the children and caregivers.

5.1 Developer experience

One of the proposed improvements was to create a personal area for the caregiver. For this comparing exercise, we will use this task as an example. This area would allow the caregiver to create and manage children records, exercises, multimedia resources and configurations, without sharing it unintentionally with other caregivers.

5.1.1 In the previous architecture

This task required some steps, from the database schema, through the models, controllers and views, first described in the old architecture:

- **Update the database schema**: it is necessary to create a relation between caregiver’s and child’s tables. As a child can have more than one caregiver (e.g. parents and therapist) and a caregiver can take care of more than one child, it is necessary to create a many-to-many association. One of the approaches is to create a “caregiverchild” table that holds both caregiver’s and child’s id (Figure 5.1).

![Figure 5.1: Many-to-many caregiver child relation](image)

The necessary code\(^1\) to create this relation in MySQL is represented in Listing 5.1.

```sql
CREATE TABLE CaregiverChild
(
    CaregiverId INT,
    ChildId INT
)
```

\(^1\) All the code shown is a simplified extract and may not work as expected.
Listing 5.1: Create table CaregiverChild

For the remaining entities (exercises, multimedia resources and configurations), it is necessary to analyze the relation between them, like it was done to CaregiverChild relation, identifying the cardinality (many-to-many, one-to-one, one-to-many, many-to-one) and how to model it in the database, and write the proper script to create the tables or columns needed.

- **Create or update the model entities:** After changing the database schema, the model entities must be updated to be consistent with it. The model is must organized in three layers (allowing to connect Struts to the database):

  - Entities classes layer
  - Data Access Object (DAO)\(^2\) layer
  - Business Object (BO)\(^3\) layer

DAO classes use Hibernate methods to query the database, while BO classes call the methods implemented in DAO classes. Both DAO and BO classes use the entities classes, which are consistent with the database using Hibernate annotations. Following the example of the CaregiverChild relation, the model entities, needed to retrieve the list of children that a caregiver takes care, should be as described in Listing 5.2.

```java
// Caregiver.java

// Attribute to relate with many Child
@ManyToMany(targetEntity = Child.class)
@LazyCollection(LazyCollectionOption.FALSE)
@JoinTable(name= "Caregiver_Child",
    joinColumns= @JoinColumn(name= "caregiverId",
        referencedColumnName="caregiverId"),
    inverseJoinColumns= @JoinColumn(name= "childId",
        referencedColumnName="childId"))
@Fetch(FetchMode.SELECT)

private List<Child> childList;
```

---

\(^2\) [https://en.wikipedia.org/wiki/Data_access_object](https://en.wikipedia.org/wiki/Data_access_object), last access 27/06/2016  
\(^3\) [https://en.wikipedia.org/wiki/Business_object](https://en.wikipedia.org/wiki/Business_object), last access 27/06/2016
// CaregiverDAO.java
// Methods to get a list of Child
public List<Child> getAllChildren() {
    setChildList();
    return getChildList();
}

public void setChildList() {
    this.childList = (List<Child>)
        getHibernateTemplate().loadAll(Child.class);
}

// CaregiverBO.java
// Method to get the caregiver's child list

public List<Child> getChildrenByCaregiver(Caregiver cg) {
    List<Child> filtered_list = new ArrayList<Child>();
    for (Child cl : cg.getChildList()) {
        filtered_list.add(getChildDao().getChildById(cl.getChildId()));
    }
    return filtered_list;
}

Listing 5.2: Caregiver and Child models

The “connection” between these classes must be declared in Spring XML configuration file (Listing 5.3).

<bean id="childDao"
    class="ChildDao"
    scope="singleton">
    <property name="sessionFactory" ref="sessionFactory" />
</bean>

<bean id="childBo"
    class="ChildBo"
    scope="singleton">
    <property name="childDao" ref="childDao" />
</bean>

<!-- Caregiver -->
Create or update actions: The controller actions must be updated (or created) to reflect the “business rule”. In this case, on Caregiver’s creation, a children list must be initialized and on child’s creation, the child must be added to the caregiver’s list (Listing 5.4).

```java
public class ManageCaregiversAction {... {
    ... // create caregiver
    public String createCaregiver() {... {
        ...
        caregiver.setChildList(new ArrayList<Child>());
    }

    // add child
    public void addChild (Child child) {
        this.childList.add(child);
    }
}
```

**Listing 5.4: ManageCaregiverAction**

The action classes must also be declared in the Spring configuration file described in Listing 5.5.
• Create or update view: to represent the results of the action, it is necessary to create or update a tile and map action results on a Struts configuration file. In this case, the caregiver sees a list of children accessing the caregiver’s module using a browser (i.e., a JSP web page, whose code is described in Listing 5.6, that will be rendered (in the server side) generating the HTML page illustrated in Figure 5.2).

Listing 5.6: JSP page

```xml
<action name="listChild" class="ManageChildrenAction"
            method="listChildren">
    <result>/jsp/adminChildren/listChildren.jsp</result>
</action>
```

Listing 5.7: Struts configuration file

The described steps must be done for children, multimedia resources, exercises and configurations, and the respective actions for creating, reading, updating and deleting them.

5.1.2 In the new architecture

Starting from the back-end, after creating the database and configuring Ebean (in the application.conf file), the first step was creating the model classes. Model classes will contain as less “business logic”
as possible (anemic domain models\(^4\)), as it will be handled by a controllers layer. Model classes extend a superclass com.avaje.ebean.Model\(^5\) which provides convenient methods for inserting, updating and deleting entity beans. An example of a model class would be the Child class (Listing 5.8).

---

```java
// Child.java
@Persistence-capable
public class Child extends Model {

@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;

private String firstName;

private String lastName;

@Column(columnDefinition = "datetime")
private Date birthDate;

private String gender;

@OneToOne(cascade = CascadeType.ALL)
@JoinColumn(name = "child_login_id")
@JsonIgnore
private Login childLogin;

@OneToMany(cascade = CascadeType.ALL)
```

\(^4\)\(https://en.wikipedia.org/wiki/Anemic_domain_model\), last access 17/04/2017

\(^5\)\(https://www.playframework.com/documentation/2.5.x/JavaEbean\) - Using Model superclass, last access 17/04/2017
private List<Sequence> sequencesList;

@OneToMany(cascade = CascadeType.ALL)
private List<PersonalMessage> personalMessagesList;
...

Listing 5.8: Entity bean - Child

Each model class should be annotated with @Entity, indicating the class as an entity bean, and have a @Id field (i.e., a primary key, although it is not mandatory). The remaining annotations are fields modifiers that will help to build the domain (e.g., @GeneratedValue defines the id field as a database identity column, @Column(columnDefinition = 'datetime') defines a specific data type, @JsonIgnore indicating that the field should be ignored by JSON parsers) or relationships between entities (@OneToMany, @ManyToMany, @OneToOne). The remaining class contains get and set methods, even though Play is able to generate them automatically. It is also necessary to create the Caregiver model, with a list of children (Listing 5.9).

// Caregiver.java
@Entity
public class Caregiver extends Model {

@Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;

private String firstName;

private String lastName;

@Column(length = 255, unique = true, nullable = false)
@Constraints.MaxLength(255)
@Constraints.Required
@Constraints.Email
public String email;

private String gender;

@OneToOne (cascade=CascadeType.ALL)
@JoinColumn(name= "caregiverLogin_id")
private Login caregiverLogin;

@ManyToMany(cascade = CascadeType.ALL)

To retrieve a caregiver's children list, it would be necessary to create a controller (represented in Listing 5.10) to perform the “business logic” and expose the method. The controller will extend a superclass play.mvc.Controller\(^6\), which provides access to some HTTP related variables, and its methods (Actions, play.mvc.Action) will return an HTTP response to the client (Result, play.mvc.Result).

```java
public class AdminChildCtrl extends Controller {
    public Result getChildren() {
        ... return ok(Json.toJson(loggedCaregiver.getChildren()));
    }
}
```

Listing 5.10: AdminChildCtrl controller

After exposing the method in the routes file, the method is ready to be used from the Caregiver module side (in Angular 2). Angular provides an HTTP module that allows making requests, which we encapsulated in a service called http-api-client.service, so we just set the common HTTP headers once and re-use it in other services. To get a caregiver’s children list, we have created a children.service with a getChildren method. Note that the domain models were mirrored in the client application (Listing 5.11).

```typescript
export class Child {
    public id: number
    public firstName: string
    public lastName: string
    public birthDate: string
    public gender: string
    public username: string
    public password: string
}

// services/children.service
@Injectable()
```

\(^6\) [https://www.playframework.com/documentation/2.5.x/JavaActions](https://www.playframework.com/documentation/2.5.x/JavaActions), last access 17/04/2017
export class ChildrenService {
  getChildren(): Observable<Array<Child>> {
    return this.http.get('/listchildren')
      .map(result => this.children = result.json());
  }
  ...
}

Listing 5.11: Child client model

The children.component will request the children list to the service and render it as described in Listing 5.12.

//components/children.component
export class ChildrenComponent ... {
  ...
  getChildren() {
    this.service.getChildren().subscribe(
      result => {
        this.children = result
      },
      err => console.log('Error loading children: ' + err)
    )
  }
  ...
}

//components/children.component.html
<div *ngFor="let child of children">
  <h4>{{child.firstName}} {{ child.lastName}}</h4>
  <p>{{ child.birthDate | date: 'dd/MM/yyyy'}}</p>
  <p>{{ child.gender }}</p>
</div>

Listing 5.12: Children component

Upon adding the CSS stylesheet, the final result is shown in Figure 5.3.

Comparing both implementation processes, it is worth mentioning:

- Play Framework and Angular2 have a modern development environment where errors are easier to understand;
- Since Play and Angular2 are more recent frameworks, it is easier to find on-line documentation, forum questions and community articles.
The previous architecture had too many configuration files, that had been replaced by only one configuration file for Play and another for Angular2;

Regarding performance, with the previous architecture, after a while using the caregiver’s application (about one hour) we had to restart Tomcat’s server. It has never happened with the Play’s server;

The abstraction level is higher in the new architecture, which allows the developer to focus on development and forget about the underneath details.

Both Play and Angular2 have a gentle learning curve.
5.2 Improvements

Regarding improvements suggested in Section 4.2, the current section describes their final result in the new version.

- **Caregiver’s personal area:** With the implemented authentication system (Section 4.1.1), the caregiver is now identified in every request and the API filters all the results by caregiver, creating his personal area, where he can keep all the children information, his own exercises and resources, without sharing it unintentionally with other caregivers (Figure 5.4).

- **CRUD operations:** In caregiver’s personal area it is possible to view, register, edit and delete children, exercises, exercises’ sequences, multimedia resources, levels and topics.

- **Preferences:** For each child, the caregiver has now a set of preference options that will help him fitting the application for each child he takes care of. For example, the messages that the animated character can utter can be defined in this area (greeting message, reinforcement message for an exercise and at the end of a sequence of exercises as shown in Figure 5.5).

![Figure 5.4: Caregiver’s personal area](image)

![Figure 5.5: Child’s personal messages](image)
• **Image exercises**: Exercises’ possible answers can be text (as it was already implemented) or images (now fully integrated with the child’s application). Each type of exercise has a specific creation form (Figure 5.6).

![Image exercises](image.png)

(a) Text exercise  
(b) Image exercise

**Figure 5.6**: Exercise forms

• **Pagination**: The exercises view shows a paginated list of the exercises where now it is possible to search for an exercise and filter by type, level or topic.

• **Forms**: The creation and edition forms are clean forms with input validation and new help hints.

• **Preview exercises**: After creating an exercise, it is possible to preview it, regardless its type (text or image answer), in the same representation as the exercise will be represented in the child application, except for the answers order, which is random in the child’s application (Figure 5.9).

• **Android version**: The application was rebuilt and it was able to run on 5.x Android versions, covering the most of the devices in the market.

• **Child’s application session**: Now the user does not need to always login since the session is kept between usages. Notwithstanding, the user can logout for his safety (e.g., if using a shared device).

Regarding the suggestions made by the caregivers who tried the previous VITHEA Kids version, the following improvements were implemented:
• **Sequences**: The exercises can now be organized into sequences (e.g., first lesson, second lesson). Inside a sequence, it is possible to sort the exercises, since it was a request from caregivers, defining the order that exercises will be presented to the child in the child’s application. This improvement was presented at the International Conference on the Computational Processing of Portuguese (PROPOR) [25].

• **Navigate through exercises**: In the child’s application, it is now possible to skip and go back inside the sequence’ exercises.
• **Animated character:** It is possible to select the preferred animated character (between Catarina, Filipe and Edgar) to be presented in the child’s application.

![Personagem animada](image)

**Figure 5.10:** Animated character options

Moreover, during the software reengineering process, it was also possible to implement the following improvements:

• **Multilingual:** The caregiver’s module was built with a multilingual support, which means that it is possible to add new languages to the system, providing a translations file for the respective language and adding it to the language menu. For now, this option is only available for caregiver’s module.

• **Gradle:** In terms of the Android project structure, we merged the two projects (Catarina’s and the exercises’ project) into one with Gradle support on Android Studio, once again aiming for developer productivity on the most recent tools.

• **Number of requests:** In the old architecture, whenever was necessary (e.g. load an exercise) a request was made to the API, which could cause a bad user experience while the user has to wait more than once. By design, it was decided that the application would request everything on the application start, showing a loading animation and avoiding making the user waits while playing the exercises. The application now makes only two requests: one for login, which only returns the authentication token (if successful), and another for the application data needed. Moreover, the only requests that will be sent are log requests (logging the child activity), which do not need to hold the user facing a loading animation since the application does not depend on them to move on.

### 5.3 New functionalities

#### 5.3.1 Prompting module

The new prompting module allows the caregiver to choose the most suitable prompting strategy for each child, as well as the prompt type. These options are in the caregiver’s Preferences menu (Figure 5.7).

---

7 [https://gradle.org/](https://gradle.org/), last access 02/05/2017
8 [http://stackoverflow.com/questions/16754643/what-is-gradle-in-android-studio](http://stackoverflow.com/questions/16754643/what-is-gradle-in-android-studio), last access 02/05/2017
9 [https://developer.android.com/studio/](https://developer.android.com/studio/), last access 02/05/2017
5.11). Regarding the strategy the caregiver can choose between “Always”, “If needed”, or “Disabled”, adapted from the prompting strategies most-to-least, least-to-most, delayed or no prompting respectively (see Section 3.2.1 and 4.3.1). Also, he can combine different types of prompts related to the size, color, legibility or visibility of the distractors.

![Figure 5.11: Child prompting options](image)

5.3.2 Reinforcement module

The new reinforcement module allows the caregiver to choose between “Always”, “At first” and “Disabled” mapping with continuous reinforcement, differential reinforcement, first attempt reinforcement and no reinforcement strategies respectively (Section 3.2.2 and 4.3.2). Also, he can choose the desired multimedia resource that will be the reinforcer itself.

![Figure 5.12: Child reinforcement options](image)

5.3.3 Emotions’ module

In the emotions’ module, the caregiver may choose to enable or disable the animated character emotions when uttering its sentences, according to the preference defined for each child. Also, we created a set of emotion related exercises (using the images collected), promoting the facial expression and emotion relation, and emotion identification.
5.3.4 Exercises set

In order to make the most of VITHEA Kids, a set of exercises’ sequences was created in a "default" caregiver’s account. These sequences are about daily tasks (e.g., “What time is it?” (Figure 5.13), “How much money do you have?”), animals (e.g., "Which animal is not a mammal?"), food (e.g., "Which aliment is a fruit?"), and emotions (e.g., "The girl is...", with an image of a girl smiling and options “happy”, “say”, “surprised”, “bored”). In the future, the goal is to allow any child to login in the VITHEA Kids application with a “guest” profile, which does not require a registration, allowing users to try the application freely. After creating interest about VITHEA Kids, users should register and use the remaining functionalities and customization options.

![Figure 5.13: What time is it?](image)

5.4 User testing

In order to assess these master thesis contributions in VITHEA Kids, the current sections described an evaluation procedure proposal for user testing. Considering that these master thesis contributions represent a major change in VITHEA Kids, an attentive evaluation process is required. This process should consist of four different tests described in the following sections.

5.4.1 Quality assurance tests

It is necessary that a group of testers (at least two people) test the basic VITHEA Kids functionalities. They should be people without previous knowledge about the application and should test it from a technical point of view, i.e., making sure that every function is implemented and there are no technical bugs.

In a first moment, the testers should navigate freely in the applications (both caregiver’s and child’s application) to explore them. In order to test the basic functionalities, some simple tasks should be performed representing VITHEA Kids’ entire flow (which is called an end-to-end test). An example of this type of test can be found [here](https://www.techopedia.com/definition/7035/end-to-end-test), last access 10/05/2017.
a simple end-to-end test would be:

- **Caregiver’s application**
  1. Register in VITHEA Kids as a caregiver;
  2. Register a child to take care of;
  3. Create one or more sequences for that child;
  4. Create one or more exercises in each sequence;

- **Child’s application**
  1. Login in child’s application with the created child;
  2. Choose a sequence;
  3. Play the sequence exercises.

The tasks should become more complex to test all the functionalities (e.g., with different prompting or reinforcement strategies, or with different preferences set).

### 5.4.2 Heuristic evaluation

The following tests phase should be done by a group of people (at least two as well) with user experience knowledge. This is called a heuristic evaluation/expert testing and it is supported by heuristics (e.g., Nielsen’s heuristics\(^\text{11}\)). These tests are not about functionality, but about how the user interface is, compared to a set of accepted usability principles.

### 5.4.3 Pilot testing

This is the most important tests phase, before testing the application with the final users. In pilot testing\(^\text{12}\), the focus is on the test procedure itself. It should be done with a group of people (at least two) unfamiliar with the application (so the testers that were present in the previous phases will not fit), and without any particular knowledge but as close as the target profile as possible (in this case a mother or father and a child without any communication disorder can fit in the role). These tests session should follow a script with well-defined tasks (the same script that will be used in final users’ tests). The main goals are to understand if the tasks are well formulated, if they are not too easy or too hard, or the session is taking too long, and to verify which data can be collected and if it can be quantified in order to generate some results (e.g., time to perform a task, number of clicks or taps, number of user’s errors, number of times that the user does not know what to do next). For the collected data, it is also

---

\(^{11}\) [https://www.usability.gov/how-to-and-tools/methods/heuristic-evaluation.html], last access 10/05/2017

\(^{12}\) [https://www.nngroup.com/articles/pilot-testing/], last access 10/05/2017
necessary to define reference values so we can assess if the application’s usability is corresponding with our expectations.

5.4.4 Usability testing

Usability tests intend to assess the application’s usability. These tests are done with the application’s final users, thus being necessary to identify who are the final users. For VITHEA Kids, there are two user groups:

- **Caregivers**: A caregiver can be a mother or father, a therapist, or someone in the family, i.e., someone who takes care of a child who possibly has some kind of difficulty in communication. The caregiver should be an adult since it is responsible for the well-being of a child (or at least someone like an older brother). He should be an interested person looking for interesting tools for helping a child to learn new behaviors, can have little or much experience with computer/mobile applications (e.g., a parent, a health or an education professional).

- **Children**: VITHEA Kids was mainly designed for children with ASD, however, it should fit for any child who has some communication disorders or who is learning some communication skills. So, for the user testing session, it would be positive to have children in different circumstances, representing each of the mentioned situations. It is hard to define a profile for the target children since each disorder has different severity levels and even inside the same disorder, there are different severity levels (as in ASD). However, it is also part of this challenge to assess if the application is customizable enough for these children.

Typically, in these tests, the user does five to ten tasks within a ninety minutes session. In our case, this could not be possible regarding children tests, reducing the number of tasks to three to five in the child’s case. The well-defined tasks should aboard the general goals of the application. The tasks for VITHEA Kids could be:

- **Caregiver’s application**
  1. Register himself in VITHEA Kids application providing his personal details;
  2. Register the child in caregiver’s personal area, proving the child’s personal details;
  3. Set the personal messages for the child;
  4. Set the child’s preferred animated character;
  5. Create a sequence called “First sequence”;

[^13]: http://www.usabilityfirst.com/usability-methods/usability-testing/, last access 11/05/2017
6. Add an exercise to that sequence with question "Which animal is not a mammal?", correct answer "Bird", and remaining options "Dog", "Cat" and "Whale".

- **Child's application**

1. Login in the child's application with the username and password created by the caregiver;
2. Choose the "First sequence";
3. Answer to the exercise's question.

In this example, the caregiver performs the main tasks of the application, however as new modules were added to the application (regarding prompting, reinforcement and emotions), the tests should aboard these too.

For the defined tasks, some reference measures could be:

- **Time**: It is expectable that the caregiver could perform the defined tasks in less than twelve minutes (total time);
- **Errors**: It is expectable that the caregiver can perform this tasks without any error (being an error some action which leads out of the context of the tasks sequence).
- **Ask for help**: It is expectable that the caregiver does not ask for help more than once.

It is harder to define these measures for the child, without knowing the condition. However we could attribute a qualitative grade to the child's experience ("very good", "good", "satisfying", "unsatisfying"), being the expectable value between "very good" and "good".

According to the tasks, more data can be collected and more measures can be defined.

In the end, the collected data should be compiled and compared to what was expected in other to assess the usability of the system.

More over, a satisfaction survey can be presented to the caregiver regarding the acceptance of the new modules. Some questions could be as following:

<table>
<thead>
<tr>
<th>How easy it was to create an exercise?</th>
<th>Very Hard</th>
<th>Hard</th>
<th>Neutral</th>
<th>Easy</th>
<th>Very Easy</th>
</tr>
</thead>
<tbody>
<tr>
<td>How easy it was to create a sequence?</td>
<td>Very Hard</td>
<td>Hard</td>
<td>Neutral</td>
<td>Easy</td>
<td>Very Easy</td>
</tr>
<tr>
<td>How useful is to upload animations?</td>
<td>Nothing Useful</td>
<td>Not Useful</td>
<td>Neutral</td>
<td>Useful</td>
<td>Very Useful</td>
</tr>
</tbody>
</table>
Nothing Useful Not Useful Neutral Useful Very Useful
How useful is to choose the animated character? □ □ □ □ □

Nothing Helpful Not Helpful Neutral Helpful Very Helpful
How helpful is to provide prompts? □ □ □ □ □

The survey answers should also be compiled and analyzed to assess the user satisfaction regarding the new functionalities.
Conclusion

Contents

6.1 Future work .......................................................... 65

63
Communication is an extremely important part of a human development. There are several disorders that affect communication directly or indirectly, namely ASD, which is a spectrum with numerous specificities. VITHEA Kids is an application that intends to help children with ASD and caregivers to fight these impairments, providing exercises that fulfill many needs. This application is free, easy to use and it is in European Portuguese, which was the main goal of its designers and developers, given the users’ needs.

However, there was space for improvement, regarding software engineering, application improvements, and the addition of new functionalities. For software reengineering process, we have implemented a RESTful API, using Play Framework for the back-end, in order to serve both caregiver and child’s applications, instead of the main project with many complex and deprecated frameworks, and another to provide web services for the child’s application. In the front-end, we have created a web application in Angular2. This process improved the developer’s experience and supported the required improvements to the applications, also improving the user experience. VITHEA Kids was also enhanced with new modules regarding prompting, reinforcement and emotions, aiming to improve the learning process of new target behaviors.

The next steps in VITHEA Kids are to perform the suggested user testing sessions, in order to get as much feedback from users as possible and continue to improve the application.

6.1 Future work

VITHEA Kids and its modules can be improved and enhanced even after this master thesis project. Some tasks left as future work are:

- Support for another multimedia resources types (e.g. video, audio, speech);
- Integration of exercises using speech recognition from VITHEA;
- Integration of the existing exercise generation module;
- During this master thesis project, the logs of the child’s application were disabled, being necessary to restore that functionality;
- Once the data is gathered it is important to show it, so a statistics module should be created;
- Creation of a dictations module, to provide dictations exercises;
• Creation of a mimic module, where the user can ask to the animated character to express an emotion' facial expression.
Bibliography


