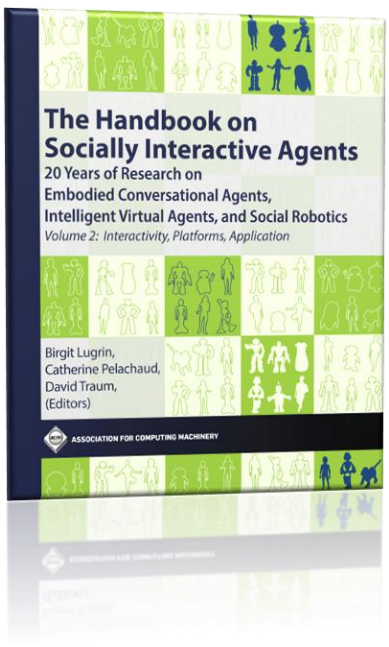
Platforms and Tools for SIA Research and Development

Arno Hartholt and Sharon Mozgai



Author note:

This is a preprint. The final article is published in "The Handbook on Socially Interactive Agents" by ACM.

Citation information:

A. Hartholt and S. Mozgai (2022). Platforms and Tools for SIA Research and Development. In B. Lugrin, C. Pelachaud, D. Traum (Eds.), *The Handbook on Socially Interactive Agents – 20 Years of Research on Embodied Conversational Agents, Intelligent Virtual Agents, and Social Robotics*, Volume 2: Interactivity, Platforms, Application (pp. 261-304). ACM.

DOI of the final chapter: 10.1145/3563659.3563668

DOI of volume 2 of the handbook: 10.1145/3563659

Correspondence concerning this article should be addressed to

Arno Hartholt, Arno Hartholt <hartholt@ict.usc.edu>, and Sharon Mozgai <mozgai@ict.usc.edu>

20 Platforms and Tools for SIA Research and Development

Arno Hartholt and Sharon Mozgai

20.1 Motivation

Developing a Socially Interactive Agent (SIA) with the goal of simulating complex human behavior in all of its intricacies and nuances is a daunting undertaking. It requires not only in-depth knowledge of individual research areas, including specialized fields within computer science, artificial intelligence, social sciences, art production, game development, and psychology, but also how these fields interconnect, both theoretically and practically. These interdisciplinary requirements go beyond the capabilities of individuals or even teams.

Fortunately, there are many tools and platforms available that researchers and developers can take advantage of, allowing us to:

- Leverage previous work, either to expand upon or to provide context for their own work.
- Contrast and compare approaches and implementations.
- Avoid starting from scratch and reinventing the wheel, reducing time and costs.
- Explore the various problem spaces within SIA collaboratively.
- Define standards, which enhance interoperability.

Research and development is becoming ever more complex. Platforms and tools allow both researchers and developers to collectively advance our understanding of and capability in the exploration and creation of SIAs in an ever more efficient and effective manner. In this chapter, we provide an overview of commonly used software solutions to the different aspects of behaviour and interaction described earlier in this handbook.

20.2 Overview

We start by discussing the history and trends of using tools and platforms, including interoperability, hardware, and distribution. Section 20.4 provides an overview of common platforms while Sections 20.5 and 20.6 delve deeper into individual tools for creating characters and character interactions, respectively. Our main focus is on Intelligent Virtual Agents (IVAs), but we briefly discuss how many of the discussed areas apply to Social Robotics (SRs) as

well in Section 20.7. We end with a discussion on current challenges, future directions, and a summary.

20.3 History and Trends

While research and development in artificial intelligence (AI) is typically acknowledged to have started in the 1950s [Crevier 1993], initial enthusiasm gave way to an "AI Winter" during the 1970s and 1980s [Hendler 2008]. In the early 1990s interest in more holistic approaches grew, with a focus on embodiment and being part of the real world, often utilizing robotics [Brooks 1991]. These techniques would be combined with real-time computer graphics and a focus on simulating human-to-human interactions to give birth to the field of embodied conversational agents [Cassell et al. 2000]. Within the context of platforms and tools we see three main trends, which we will explore below:

1. *Interoperability*: from relatively isolated research efforts toward broader collaboration and the development of standards.
2. *Hardware platforms*: from supercomputers to desktops to multiplatform solutions, including mobile, web, Augmented Reality (AR) and Virtual Reality (VR).
3. *Distribution*: from dedicated research websites and propriety software to standard open source sharing platforms and commercial web services.

Initial *interoperability* efforts were often isolated and focused on a subset of SIA research fields. Early examples include CMUSphinx [Lamere et al. 2003] for automated speech recognition, BEAT [Cassell et al. 1994, 2001] for nonverbal behavior generation, SPUD [Stone and Doran 1997], CSLU Toolkit [Sutton et al. 1998], and TRINDI Dialogue Move Engine Toolkit [Larsson and Traum 2000] for natural language processing, and Festival [Taylor et al. 1998] for text-to-speech generation. These isolated efforts made collaborative research, integration, and sharing challenging, in particular because approaches would not align [Gratch et al. 2002]. As a result, standards were proposed and iterated upon. Examples include Knowledge Query and Manipulation Language (KQML) [Finin et al. 1994], Speech Synthesis Markup Language (SSML) [Taylor and Isard 1997], Virtual Human Markup Language (VHML) [Marriott 2001][1], Affective Presentation Markup Language (APML) and Discourse Plan Markup Language (DPML) [De Carolis et al. 2002], MPEG-4 facial animation [Pelachaud 2002], Avatar Markup Language (AML) [Kshirsagar et al. 2002], Multimodal Utterance Representation Markup Language (MURML) [Kranstedt et al. 2002], Character Markup Language (CML) [Arafa and Mamdani 2003], Artificial Intelligence Markup Language (AIML) [Wallace 2003], Web3D Consortium's HAnim [Web3D 2006], and Behavior Markup Language (BML) [Kopp et al. 2006]. Out of these, SSML and BML are the main standards still in use

[1] The website https://www.vhml.org is a wonderfully preserved artifact of the early aughts and the authors encourage any interested reader to take a trip down memory lane.

today. SSML allows a character's utterance to be marked up to indicate how its speech should be generated. BML describes the nonverbal behavior for a character and is based on a large academic collaboration of integrating previous standards. It is part of the SAIBA framework [Kopp et al. 2006], which also includes the Function Markup Language (FML) [Heylen et al. 2008b]. Most of these are not true standards in the traditional sense of the word; they have not been ratified by official bodies, but instead are common formats often used by researchers in the field of SIA. For more details on multimodal interaction architectres, see Chapter 16 "The Fabric of Socially Interactive Agents: Multimodal Interaction Architectures" [Kopp and Hassan 2022] of this volume of this handbook.

For *hardware platforms*, computing power was still relatively scarce around the turn of the millennium, which led to either basic applications or the need for large, distributed computing servers [Rickel et al. 2001] with specialized and proprietary functionality and content [Rickel et al. 2002]. Increasingly powerful desktop systems resulted in ever more powerful applications being able to run on personal computers [Swartout et al. 2006]. The web has seen early adaption of SIAs [André et al. 1998, Evers and Nijholt 2000, Noma et al. 2000] and supporting tools [Bickmore et al. 2009]. Smartphones and tablets have seen a range of SIAs [Bickmore et al. 2010, Doumanis 2013] as well as tools to support its development [Feng et al. 2015, Klaassen et al. 2012]. Finally, a sizable effort is currently focused on AR and VR [Hartholt et al. 2019a, Holz et al. 2011]. The move toward ever more personalized, pervasive, and immersive computing devices has resulted not only in the democratization of computing, but also in a proliferation of tools that support the development of SIAs, in particular the use of game engines (e.g., Unity, Unreal Engine), see Section 20.4.2.

Early SIA *distribution* methods often relied on providing software binaries or source code through university or personal websites, followed by centralized services that include version control, bug tracking and documentation, predominately SourceForge in the 2000s [Howison and Crowston 2004] and GitHub in the 2010s and beyond [Kalliamvakou et al. 2014]. While the open source philosophy initially was mainly supported by researchers and a small number of companies, open sourcing software and data is becoming increasingly commonplace, with traditionally closed companies opening up portions of their IP in order to leverage the advantages of community-based development, including Microsoft[2], Apple[3], Facebook[4], Epic Games[5] and Unity[6]. At the same time, the proliferation of personal assistants and related "AI" capabilities has resulted in turning previously challenging areas into commodity technologies (e.g., speech recognition, text-to-speech). These are often available in the form of online

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

[3] https://github.com/apple

[4] https://github.com/facebook

[5] https://www.unrealengine.com/en-US/ue4-on-github

[6] https://github.com/Unity-Technologies

services, offering researchers and developers a range of interconnected capabilities they can leverage in designing and developing SIAs.

These three trends have led to the democratization of many capabilities, which in turn leads to more technical solutions, increased competition, and improved accessibility.

20.4 Agent Platforms

We define a platform as having a suite of capabilities, which are integrated in a principled manner, that together cover several required features of an SIA, and that should be extendable. Based on these criteria, we define three categories of platforms, which we will explore below:

1. *Cognitive architectures*: principled approaches to simulate aspects of the human mind.
2. *Commercial platforms*: privately developed game engines to create animated characters and their behaviors.
3. *Academic platforms*: integrated systems built by research organizations that cover many SIA-specific capabilities.

20.4.1 Cognitive Architectures

Cognitive architectures typically only cover the mind of an intelligent agent, but many have incorporated perception of and acting in the world as explicit notions. This makes them suitable candidates to integrate them with SIAs. We will cover several of the more commonly used architectures from a practical point of view. For a more in-depth discussion of cognitive architectures, see Chapter 16 on "The Fabric of Socially Interactive Agents: Multimodal Interaction Architectures" [Kopp and Hassan 2022] of this volume of this handbook.

Soar is one of the early cognitive architectures, designed and developed by Allen Newell, John Laird and Rosenbloom in 1983 [Laird 2012]. Soar is a general cognitive architecture for developing systems that exhibit intelligent behavior using symbolic reasoning. It is goal oriented and uses operator rules within a problem space to select its next action which affects the overall state. It includes procedural, semantic and episodic memory, that work together with short-term memory and learning mechanisms. Soar started off using general-purpose mechanisms, but later versions incorporated dedicated modules to better support specialized functions, including emotions and visual sensing. It has been used to develop intelligent conversational agents [Rickel et al. 2001, Swartout et al. 2006] and robotics [Laird et al. 2012]. Soar can use several external programming languages, including C++, Java, Python, and TCL, through the Soar Markup Language (SML). It includes a suite of supporting development tools, including visual and command line tools. It is available under a modified BSD 2-clause license. See https://soar.eecs.umich.edu for more details.

ACT-R was originally developed by John Anderson [Anderson et al. 1997] who was heavily influenced by Allen Newell and Soar. ACT-R uses symbolic reasoning, declarative and procedural memory, and perception and motor modules to interface with the world.

Production rules fire when matching certain states which in turn affect the overall state of the system. ACT-R very explicitly bases its cognitive assumptions on results derived from psychology experiments and allows researchers to collect quantitative measures that can be directly compared with similar measures obtained from human participants. It has been used most notably in an intelligent match tutor [Ritter et al. 2007]. It is developed in Lisp and can interface with Python and other languages using JSON. ACT-R is licensed under LGPL v2.1. For more details, see http://act-r.psy.cmu.edu.

OpenCog was initially released in 2008 by Ben Goertzel and David Hart [Hart and Goertzel 2008]. It aims to achieve general intelligence through tightly integrated cognitive algorithms, a strategy dubbed cognitive synergy. OpenCog includes native support for natural language processing, reasoning and inference, embodiment, and psychological states. It includes a symbolic knowledge representation that multiple cognitive processes can work on, called AtomSpace, which is a graph-based knowledge representation database with a query and reasoning engine. Applications have been developed both for IVAs and robotics. OpenCog uses C++, Python, and a custom language in support of AtomSpace called Atomese. It is available under the AGPL 3 license (see https://opencog.org).

TinyCog was released in 2015 by Frank Bergmann and Brian Fenton [Bergmann and Fenton 2015], following the tradition of Soar and ACT-R. It specifically aims to provide a minimalist implementation of a cognitive architecture, which makes it a good starting place for newcomers, even though it is no longer in active development. TinyCog uses what they call Scene Based Reasoning, in which scenes represent real world environments on which plans can be executed. The main focus is on robotics with implementations for perception, actions, and learning. TinyCog is written in ProLog, licensed under GPL v3, and can be found at http://tinycog.sourceforge.net.

Sigma (Σ) is one of the more recent efforts and is headed up by Paul Rosenbloom and Volkan Ustun [Rosenbloom et al. 2016]. Its design and development is driven by four goals: grand unification, generic cognition, functional elegance, and sufficient efficiency. It aims to achieve this by combining traditional cognitive architecture concepts with the use of factor graphs [Kschischang et al. 2001] as the main underlying mechanism. Sigma primarily targets IVAs and has developed several sample projects that include perception, reasoning, learning, emotions, and natural language processing. Sigma is written in Lisp and available under the BSD 2-clause license at https://cogarch.ict.usc.edu.

20.4.2 Commercial Platforms

There are many individual commercial tools available for use within SIA, which we will cover in Sections 20.5 and 20.6, yet not many offerings are available that combine these into platforms specifically in support of SIA development. However, game engines do provide a range of integrated functionality, in particular for IVAs. Modern game engines do not only cover rendering, but also animation, sound, networking, and so on. They typically cover

multiple hardware platforms (e.g., mobile, web, AR, VR) and offer flexible development environments. While they are mainly focused on game development, they are general purpose tools that are very useful in creating IVAs.

The two most popular game engines are *Unity* and *Unreal Engine*. Unity started off as a game engine for beginners and smaller developers, while Unreal Engine targeted large, professional teams. As a result, Unity is easy to pick up, allows for rapid iteration, has a large community and asset store, and excellent multiplatform support. Unreal Engine on the other hand has a longer history and shines in creating higher fidelity graphics—at the cost of an increased learning curve—and is completely open source. Unreal Engine and Unity are in fierce competition with each other, which ultimately benefits researchers and developers. Both engines are now free for academic use and small developers. Unity continues to catch up in terms of visual fidelity, while Unreal Engine provides more and more functionality to streamline development. Both typically require at least some level of programming to create IVAs. We will discuss each game engine in more detail below. Afterwards, we point toward several alternatives.

20.4.2.1 Unreal Engine

Figure 20.1 Siren character created by Epic Games in Unreal Engine in 2018, together with Cubic Motion, 3Lateral, Tencent, and Vicon. With permission of Epic Games ©2021.

Unreal Engine is developed by Epic Games, who started licensing it in 1996. Epic Games, unlike Unity, develop their own games in addition to the engine. The most well-known of these are Unreal, Gears of War, and Fortnite. Unreal Engine can be obtained at https:

//www.unrealengine.com and is free for non-commercial use. Access requires an account, including for use of the source code.

Unreal Engine is a powerful game engine used for many high-fidelity video games on PC and consoles. It offers all of the main required functionality out of the box, including networking, GUIs, animation, physics, and audio. Unreal Engine no longer has a scripting language, but instead uses either C++ or a visual scripting language called Blueprints Visual Scripting. This approach is one indication of Unreal Engine traditionally focusing on large, professional teams, where programmers would do the core development and create tools and templates for designers.

While Unreal Engine supports most common platforms, including, mobile, web, AR and VR, the implementation is typically less robust and user-friendly than Unity. The Unreal Engine Editor offers many graphical interfaces for most areas, though, and usability is improving. There are many tutorials available, including a dedicated section for developers transitioning from Unity.

Unreal Engine particularly shines in graphical fidelity (see Figure 20.1). It offers powerful tools and shaders in order to fully customize the look and feel of the environment, characters, and objects, which has made it a successful tool for non-gaming application, including architecture, previz production, and advertisement. Unreal Engine now offers easy to create, high-fidelity characters through its MetaHuman tool, although using these as conversational agents is as of yet nontrivial. Epic Games' massive success with Fortnite has resulted in solid multiplatform and multiplayer capabilities, as well as enough income to grow the engine. Unreal Engine is released several times a year, with version 5 released in 2022

20.4.2.2 Unity

Unity can be obtained at https://unity.com through a subscription model. There is a free version for students and individuals, with professionals and larger teams paying a monthly fee. All versions are technically equivalent, with the main differences lying in online features, team communication features, technical support, and available assets. Unity has open sourced portions of their product at https://github.com/Unity-Technologies, but the source code of the main engine, written in C/C++, can only be obtained through a paid license.

The Unity Editor is a development environment that allows visually manipulating game objects that can have C# scripts associated with them. This enables a range of developers with different levels of technical expertise, including designers, artists and programmers, to be productive. Unity supports graphical user interfaces (GUIs), audio, networking, pathfinding, and character animation out of the box for both 2D and 3D projects. Its animation system, originally called Mecanim, is quite powerful, offering graphical state machines and blending parameters. It offers solid tutorials and has a wide community, which—while varying in skill level—offers lots of help as well as scripts and assets through the Unity Asset Store. Unity itself offers many online services, for example cloud build solutions for all platforms

Figure 20.2 Character created in Unity 2022.1 as part of their promotion video, Enemies. Image by Unity Technologies. Source: https://blog.unity.com/news/introducing-enemies-the-latest-evolution-in-high-fidelity-digital-humans-from-unity.

it supports, which includes Windows, Mac, Linux, iOS, Android, WebGL, and major game consoles. Unity's robust multiplatform support has resulted in 69% to 91% of VR and AR applications being developed with Unity [Marvin 2018].

Looking toward the future, Unity has implemented machine learning tools as a separate package, called ML-Agents[7]. This allows for reinforcement learning (based on TensorFlow[8]), imitation learning, and other methods using a Python API, with Unity visualizing agents and their environments. Unity keeps advancing their rendering pipelines in order to improve graphical fidelity and to provide developers with more control. See Figure 20.2 for one of their demo characters. Unity is also working on enhancing performance by moving from an object-oriented to a data-oriented design of the core engine, an approach it has called Data-Oriented Technology Stack (DOTS). DOTS uses a range of techniques, one of which is the Entity-Component-System (ECS) architectural pattern, which decouples aspects of real-time simulation (e.g., graphics, physics, AI) in order to process entities more efficiently and to support safe multithreading.

Starting in 2020, Unity releases major new versions two times a year (e.g., 2020.1, 2020.2) and supports a final yearly release for up to 2 years (e.g., 2020.3). A unity roadmap can be found at https://unity3d.com/unity/roadmap.

[7] https://github.com/Unity-Technologies/ml-agents

[8] https://www.tensorflow.org

20.4.2.3 Honorable Mentions

There are numerous other game engines available, of which we will briefly discuss a selection.

Id Tech (https://github.com/id-Software) is a series of game engines by Id Software that started after one of the first popular 3D games, Wolfenstein, with Doom (1993) and Quake (1996). Older versions up till Id Tech 4 are released under the GPL license.

CryEngine (https://www.cryengine.com) is a high-fidelity game engine, that was originally released in 2002. It has lost some popularity in recent years, partly due to a lack of funding for continued development. It is still used for professional game development and aims to make a comeback. Part of this strategy is to release the engine open source (https://github.com/CRYTEK/CRYENGINE), freely available for non-commercial use.

Panda3D (https://www.panda3d.org) was originally created by Disney and made open source in 2002. It is written in C++, uses Python, supports multiple platforms, and is currently available under the BSD license.

The *Source Engine* (https://developer.valvesoftware.com/wiki/SDK_Installation) was originally released in 2004. It is created by Valve, co-creator of the HTC Vive VR headset. It is not used much beyond Valve games itself (e.g., Half Life, Counter Strike, Team Fortress, DOTA 2) and only offers an SDK rather than the full game engine, but recent VR interest may change this.

Godot (https://godotengine.org) is relatively recent, with an original release date of 2014, and aims for smaller projects. It offers a user-friendly development environment, supports C++ and C#, is multiplatform, and is open source under the MIT license.

Lumberyard (https://aws.amazon.com/lumberyard) is developed by Amazon and released in 2016. It uses CryEngine as a foundation, maintaining its high-fidelity approach while aiming more broadly at developers to create virtual worlds rather than just games. It integrates with both Twitch—a popular game streaming service—and Amazon's AWS cloud services. Its source code is available at https://github.com/aws/lumberyard.

Amazon Sumerian (https://aws.amazon.com/sumerian) is an online web authoring tool for creating web, AR, and VR experiences that includes interactive characters, originally released in 2017. It integrates with many of Amazon's own services in order to cover a range of SIA related capabilities, including speech recognition, natural language processing, nonverbal behavior generation and text-to-speech. The development is web-only, using a visual editor and JavaScript, and as such not very customizable or expandable. It can, however, be combined with 3rd party software [Monteiro and Pfeiffer]. Amazon Sumerian requires a monthly subscription, based on the level of services used.

20.4.3 Academic Platforms

Few academic organizations possess the expertise and funding to create platforms that incorporate all aspects of SIA. As a result, platforms typically slowly emerge over time, based on previous work and collaborations, with varying levels of continued maintenance and support.

We discuss here two of the main SIA platforms, *Greta* and the *Virtual Human Toolkit*, chosen because of their lengthy history, broad coverage of SIA areas, extendability, and ongoing use and support. Afterwards, we point toward several alternatives.

20.4.3.1 Greta

One of the earliest and most fleshed-out SIA platforms is *Greta*, originally released around 2005 [Poggi et al. 2005]. It grew out of earlier work that focused on the importance of gaze in coordinated verbal and nonverbal communication [Poggi et al. 2000], the design of a reflexive agent capable of showing and hiding emotions [De Carolis et al. 2001], and the creation of a detailed 3D face capable of showing nuanced facial expressions as well as detailed facial and skin deformation, including wrinkles [Pasquariello and Pelachaud 2002]. It is most notable for being able to modulate verbal and nonverbal behavior based on personality and other factors as well as including automated listening behaviors and backchanneling [Bevacqua et al. 2010].

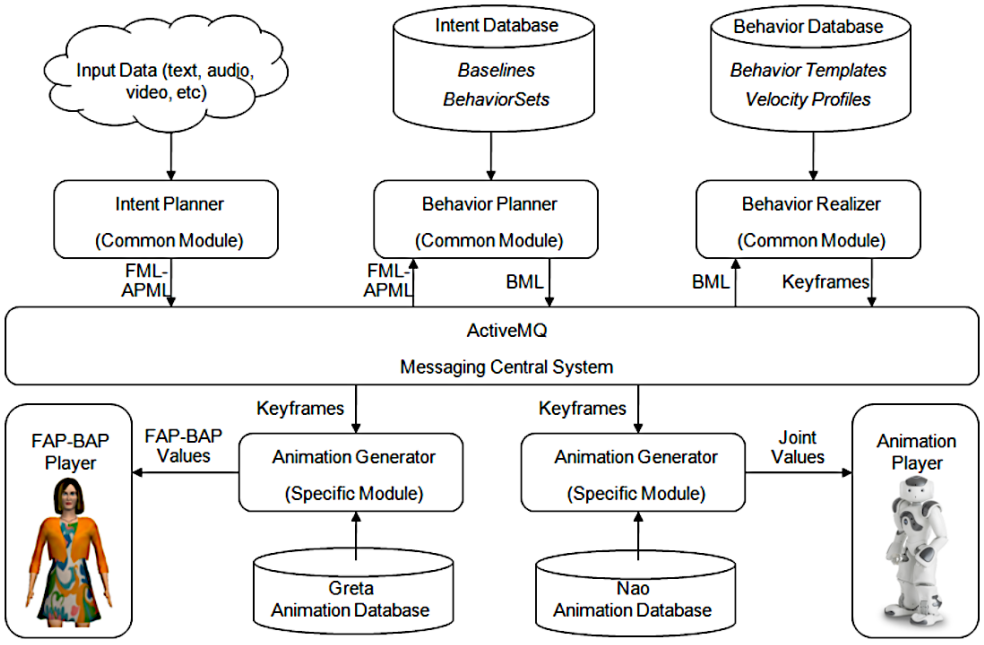


Figure 20.3 The Greta architecture, adapted from its GitHub page.

The architecture of Greta is modular and follows the publish and subscribe whiteboard approach [Niewiadomski et al. 2009]. It uses either the Psyclone messaging system [Thórisson et al. 2005] or ActiveMQ [Snyder et al. 2011]. It is compliant with the SAIBA framework [Vilhjálmsson et al. 2007], developed in Java, and extendable with custom modules. Its main platform target is Windows, with versions available for mobile and web as well. See Figure 20.3 for an overview of the architecture.

Greta can interface with a range of external audio-visual sensing and speech recognition systems, including Watson [Morency et al. 2005], PureData[9], and SSI [Wagner et al. 2013]. These signals are processed by the Intent Planner, which outputs a communicative intent in FML-APML [Heylen et al. 2008a]. The Behavior Planner takes this as input and creates a behavior schedule in BML [Kopp et al. 2006]. Finally, the Behavior Realizer creates the actual movement of the agent, which can be done in the FAP-BAP Player using the MPEG-4 standard [Ostermann 2002], a robot (e.g., Aibo) [Niewiadomski et al. 2009], or a game engine, including Ogre3D or Unity. Text-to-speech is provided by either MaryTTS[10] or CereVoice [Aylett and Pidcock 2007].

Many research efforts have used Greta, including the SEMAINE project, which aimed to develop an Sensitive Artificial Listener [Schroder et al. 2011]. The HUMAINE project focused on the emotional aspects of human to agent interactions [Petta et al. 2011], which led to the creation of the HUMAINE Database that contains clips of the use of emotion in everyday interactions [Douglas-Cowie et al. 2011]. The TARDIS project created a framework for developing agents who could offer social coaching within the context of job interviewing [Anderson et al. 2013]. Ask Alice is an example of the ARIA project (Artificial Retrieval of Information Agent) in which a user can interact with a virtual Alice from Wonderland [Valstar et al. 2016].

Greta is available at https://github.com/isir/greta under a mix of LGPL v3 and GPL v3 licensing. It provides documentation and tutorials as well as an overview of associated projects.

20.4.3.2 Virtual Human Toolkit

The *Virtual Human Toolkit* (VHToolkit) is a convergence of approaches and technologies researched and developed at the University of Southern California Institute for Creative Technologies (ICT), released in 2009 [Hartholt et al. 2013]. The MRE [Rickel et al. 2001] and SASO projects [Swartout et al. 2006] provided the overall architecture and nonverbal behavior technology, combined with natural language processing and rendering technologies from the SGT Star project [Artstein et al. 2008], which were integrated into a common platform as part of the Gunslinger project [Hartholt et al. 2009].

The VHToolkit follows the SAIBA framework [Vilhjálmsson et al. 2007] and has a modular architecture. Modules mainly communicate with each other through message passing using a custom protocol called VHMsg, developed on top of ActiveMQ [Snyder et al. 2011], see Figure 20.4, where regular arrows indicate messages and bolded arrows direct connections. Modules can be written in a range of languages, the most common of which are C#, Java and C++. This allows relatively easy incorporation of new modules as long as

[9] http://puredata.info

[10] http://mary.dfki.de

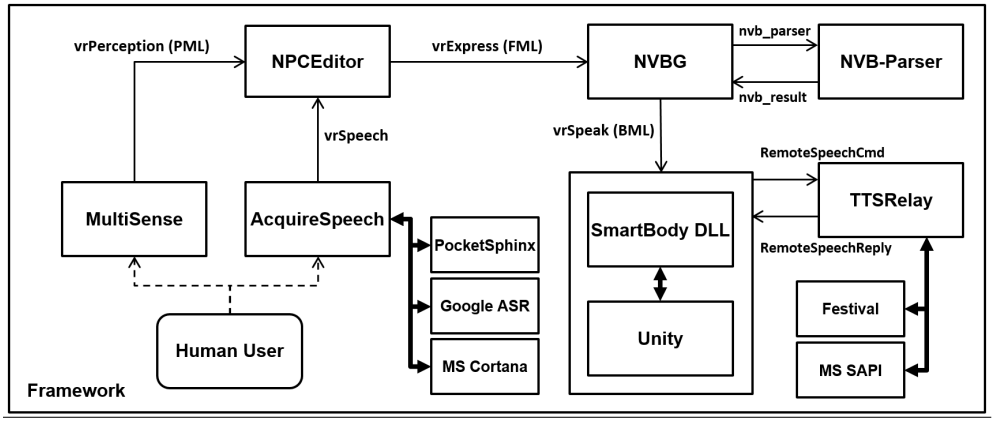


Figure 20.4 The Virtual Human Toolkit architecture, adapted from its website.

they adhere to the VHMsg protocol. The VHToolkit mainly supports Windows, although a multiplatform version is in internal development [Hartholt et al. 2020].

The default speech recognition solution is PocketSphinx [Huggins-Daines et al. 2006], with options for Google ASR and native Windows 10. Audio-visual sensing is provided by MultiSense, which is built on top of SSI [Wagner et al. 2013]. MultiSense combines sensing producers and consumers to provide sensing and behavioral information to the rest of the system using Perception Markup Language (PML) messages [Scherer et al. 2012]. Natural language processing is provided by the NPCEditor, a statistical text classifier that matches novel user input to the best character response output [Leuski and Traum 2011]. It can take custom Groovy scripts to provide dialogue management functions. The NPCEditor sends FML to the NonVerbal Behavior Generator (NVBG), which generates a BML schedule based on syntactic and semantic rules [Lee and Marsella 2006]. This BML schedule is sent to SmartBody, a procedural character animation and simulation platform and one of the most powerful BML realizers available [Shapiro 2011]. Text-to-speech defaults to Festival [Black et al. 1998], with options for CereVoice [Aylett and Pidcock 2007] and MS SAPI [Shi and Maier 1996]. Rendering is provided by Unity.

The VHToolkit has formed the core of many virtual human prototypes, both for research as well as deployment. ELITE and INOTS combine virtual humans with intelligent tutoring to offer a system that allows young officers in the US Army and Navy to learn and practice leadership and counseling skills [Hays et al. 2012]. The Museum Guides were a lifesized exhibition at the Boston Museum of Science, providing kids with information about the museum and STEM topics [Swartout et al. 2010]. PAL3 is a meta-tutor that will accompany a student or professional throughout their career, providing advice and recommendations along the way [Swartout et al. 2016]. New Dimensions in Testimony allows museum visitors to

interact with a lifesized capture of a Holocaust survivor [Traum et al. 2015]. VITA allows young adults with autism to practice job interviews [Burke et al. 2018]. It has been expanded to also serve veterans transitioning back into society as well as juveniles [Hartholt et al. 2019c]. An AR prototype was developed for the Magic Leap AR headset [Hartholt et al. 2019b]. The Battle Buddy is a mobile passive sensing agent designed to collect multimodal data with passive sensors native to popular wearables (e.g., Apple Watch, Fitbit, and Garmin) as well as through user self-report. It delivers personalized and adaptive multimedia content via smartphone application specifically tailored to the user in the interdependent domains of physical, cognitive, and emotional health [Mozgai et al. 2020].

The VHToolkit is free for academic use and can be obtained at https://vhtoolkit.ict.usc.edu. This website includes documentation, tutorials and a forum.

20.4.3.3 Honorable Mentions

Relational Agents are web-based 2D IVAs often used within the healthcare domain [Bickmore et al. 2009]. The framework offers a task planner and dialogue manager with associated ontology [Bickmore et al. 2011], a web BML realizer, and text-to-speech integrations. It uses Java and Flash and is available at https://relationalagents.com/demo/index.html.

WASABI aims to combine physical emotion dynamics with cognitive appraisal in order to simulate infant-like primary emotions as well as cognitively elaborated secondary emotions [Becker-Asano and Wachsmuth 2010]. The source code is freely available under the LGPL v3 license through https://www.becker-asano.de/index.php/research/wasabi.

The *Virtual People Factory* is a web authoring and runtime platform, often used to create virtual patients [Rossen and Lok 2012]. The website can be accessed at http://virtualpeoplefactory.com.

Visual SceneMaker is an authoring tool that allows non-experts to create interactive presentations [Gebhard et al. 2012]. It has been used in many projects, including TARDIS [Anderson et al. 2013] and is freely available at https://github.com/SceneMaker/VisualSceneMaker.

ADAPT focuses on full body animation, navigation and object interaction by combining a range of different approaches and technologies, including SmartBody, into a single framework [Shoulson et al. 2013]. It can be obtained at https://github.com/ashoulson/ADAPT.

The *Articulated Social Agents Platform (Asap)* provides a collection of software modules for both IVAs and SRs [Kopp et al. 2014]. Asap is SAIBA compliant and includes Flipper for dialogue management [Ter Maat and Heylen 2011]. Its main language is Java and is available under the LGPL v3 license at https://github.com/ArticulatedSocialAgentsPlatform/Asap/wiki.

The *Generalized Intelligent Framework for Tutoring (GIFT)* is mainly focused on providing intelligent tutoring capabilities both on desktop and the web [Sottilare et al. 2017]. It provides some SIA capabilities through integration with a subset of the VHToolkit. It is open source and accessible at https://www.gifttutoring.org.

M-PATH focuses on empathetic conversations, using SmartBody [Shapiro 2011] in combination with a custom dialogue manager [Yalçın and DiPaola 2019]. It is hosted at https://github.com/onyalcin/M-PATH.

The *Standard Patient Studio* allows doctors and medical students to create and practice with their own standard patients [Talbot and Rizzo 2019]. It is an online authoring tool that starts with an interactive, healthy patient as a baseline to deviate from. It includes student feedback capabilities and is accessible at https://www.standardpatient.org.

20.5 Tools to Create Appearance and Nonverbal Behavior

For a digital character to be rendered, it first needs to be modeled, which includes creating the mesh (i.e., the shape of the character) as well as the textures (i.e., the paint on that shape). The mesh needs to be rigged to a skeleton, which contains all the joints that can be animated (e.g., legs, fingers, jaw). The animated skeleton drives the deformation of the mesh based on skinning information. This joint-driven animation approach can be used for both the body and the face. In addition, the mesh can be deformed directly by creating a series of blend shapes—also called morph targets—which are detailed poses of a portion of the mesh (e.g., the mouth or cheeks) that can be blended together to create the desired effect (e.g., a smile, bulging biceps). When all these aspects are integrated, this source art can be exported to a game engine, typically in the FBX format[11].

In this section we will discuss individual tools that cover any aspect of the appearance and nonverbal behavior of an IVA, including *modeling* and *animating* a character as well as *generating and realizing nonverbal behavior*.

20.5.1 Modeling

Modeling a character includes both creating the mesh (i.e., shape) and textures (i.e., paint) of the character. Traditional modeling software allows artists to create the character mesh as well as the UV-layouts that indicate how a 2D texture should be mapped to the 3D mesh. Maya[12] and 3DMax[13] by Autodesk are often used by professionals, while Blender is a free and open source alternative[14], and Houdini offers a free version[15]. These programs typically contain the overall character rig, with all the necessary elements to be exported to the game engine. Textures can be created with general purpose tools like Adobe Photoshop[16], the open

[11] https://www.autodesk.com/products/fbx

[12] https://www.autodesk.com/products/maya

[13] https://www.autodesk.com/products/3ds-max

[14] https://www.blender.org

[15] https://www.sidefx.com/products/houdini

[16] https://www.adobe.com/products/photoshop.html

source alternative Gimp[17], or with specialized software, including Mari[18] and Substance Painter[19]. Specialized sculpting software, including Mudbox[20] and ZBrush[21], allows artists to sculpt and paint high-detail models that can be exported into Maya, 3DMax or Blender. The mesh typically gets downsized to contain less detail, while many of the original details gets maintained in the textures. This technique allows for the simulation of detail while maintaining performance. Upcoming advances in game engine technology may fully automate this process, retaining all details without the need for low-fidelity meshes with high-fidelity texture maps.

Scanning real people in order to digitize their appearance is becoming ever more feasible, either through commodity hardware [Achenbach et al. 2017, Chibane et al. 2020, Shapiro et al. 2014] or through specialized hardware like the USC ICT LightStage [Debevec et al. 2000]. This process consists of creating a series of photographs of the subject, either full body or just the face, to then stitch together using photogrammetry into a 3D mesh with textures. While commodity hardware typically results in a character that has the original lighting conditions baked in, high-end solutions like the LightStage can capture images under a myriad of lighting conditions, allowing them to re-light the resulting 3D character in any novel environment. More recent efforts focus on generating models from a single image, for example *Tex2Shape* [Alldieck et al. 2019], available together with related tools at https://virtualhumans.mpi-inf.mpg.de/software.html.

Character art assets can also be created or obtained directly from 3rd party sources, including Autodesk Character Generator[22], iClone[23], MakeHuman[24], Mixamo[25], Renderpeople[26], TurboSquid[27], Unity Asset Store[28], and Unreal Marketplace[29].

Ultimately, any of these assets will need to be run in real-time in a game engine, combining skeleton, mesh, textures, and blend shapes. In addition, modern characters require materials and shaders in order to provide realistic reflections of light on different surfaces (e.g., cloth,

[17] https://www.gimp.org
[18] https://www.foundry.com/products/mari
[19] https://www.substance3d.com/products/substance-painter
[20] https://www.autodesk.com/products/mudbox
[21] https://pixologic.com
[22] https://charactergenerator.autodesk.com
[23] https://www.reallusion.com/iclone
[24] http://www.makehumancommunity.org
[25] https://www.mixamo.com
[26] https://renderpeople.com
[27] https://www.turbosquid.com
[28] https://assetstore.unity.com
[29] https://www.unrealengine.com/marketplace

skin, eyes) as well as physics on cloth and hair. Tools like Substance Designer[30] and PhysX[31] can support these.

20.5.2 Animation

Animating IVAs brings with it a set of particular challenges compared to offline animation for movies or even real-time animation for video games. IVA characters need to be able to respond dynamically and in real-time to the user, which requires lip-synching to match the generated character speech, conversational gestures in support of the words and overall meaning, as well as for body language, facial expressions and gaze to match the underlying communicative intent, personality, and cognitive processes. For use with nonverbal behavior generation and realization (see below), they will need to be annotated with metadata that indicates the timing of the phases of the animation so that it can be synchronized with the character's speech.

Animation techniques can be divided into traditional *keyframe animation*, *motion capture* (mocap), and *procedural animation*.

With *keyframe animation*, an artist uses an animation rig to create a series of poses—the keyframes—that an animation system then blends together to create the final performance. This can be labor intensive, but allows the artist complete control over the final result. This is typically done in tools like Maya, 3DMax or Blender.

Mocap maps an actor's movements onto a digital character. High-end mocap studios use dedicated sets with specialized cameras that look at markers on a suit that the actor wears. This typically gets processed with tools like MotionBuilder[32] before it's used in the rest of the animation pipeline. Markerless suits can be used outside of expensive studios, for example, Rokoko[33]. More commodity hardware like webcams or 3D depth cameras can be used for a lower-cost (and typically lower quality) solution, for example f-clone[34] and iPi Soft[35].

Finally, animations can be *procedurally generated*, either using math functions [Lee et al. 2007] or example-based controllers, where existing animation data forms the basis for further manipulation or blending to get the desired performance [Shapiro 2011].

Regardless of the process, animations can be purchased from 3rd party sources, typically from the sources mentioned in the Modeling section above.

While the above approaches can broadly apply to both the body and the face, the latter typically requires special attention. Facial animation is complex, layering and blending lip-synching with facial expressions and gaze. Most systems base the facial expressions on the Facial Action Coding System (FACS) framework [Ekman 1997]. Lip-sync tools like

[30] https://www.substance3d.com/products/substance-designer

[31] https://www.geforce.com/hardware/technology/physx

[32] https://www.autodesk.com/products/motionbuilder

[33] https://www.rokoko.com

[34] http://f-clone.com

[35] http://ipisoft.com

FaceFX[36] or dedicated Unity or Unreal Engine plugins can generate a viseme schedule (i.e., individual base units of mouth shapes) based on the character's phoneme schedule (i.e., individual base units of speech sounds), either by analyzing an existing audio file or by obtaining a phoneme schedule in real-time from a text-to-speech provider. Full facial performance capture uses cameras to track an actor's face and translate the performance to a digital character, for example using Blender[37], FaceWare[38], and Unreal Engine[39].

20.5.3 Nonverbal Behavior Generation and Realization

Many traditional IVAs are developed using the SAIBA framework [Kopp et al. 2006], where an agent "mind" generates a communicative intent in the form of FML, which gets translated into BML by a generator, which in turn gets realized in an animation system and renderer. BML describes at a high level what a character should do (e.g., speak words, gaze at object, gesture, etc.) and how to synchronize all behaviors. For instance, it can describe that the emphasis point of a conversational gesture coincides with the pronunciation of a specific word. The BEAT system was instrumental in laying the groundwork for this approach [Cassell et al. 2004]. For more details on multimodal interaction architectures, see Chapter 16 on "The Fabric of Socially Interactive Agents: Multimodal Interaction Architectures" [Kopp and Hassan 2022] of this volume of this handbook.

There are several generators available that take FML as input and produce BML as output. The Greta platform (see Section 20.4.3.1) provides the *Behavior Planner*. It takes the high-level intent of the agent and generates through rules a series of associated nonverbal behavior intents, which can be based on role, personality, and context. This was first described in [De Carolis et al. 2002] as the APML, one of the efforts on which FML is modeled. The VHToolkit includes the *NVBG* [Lee and Marsella 2006] a rule-based system that analyzes a character's surface text semantically through keyword scanning and syntactically with the Charniak parser [McClosky et al. 2006]. A series of rules fire to generate head gaze, nods and shakes, and to select conversational gestures and facial expressions. The resulting schedule gets pruned based on rule priorities when competing overlapping behaviors are triggered. Defaults can be overwritten per character or personality.

BML realizers take the high-level behavior schedule and are responsible to execute this in real-time, synchronizing all requested behaviors. Greta includes the *Behavior Realizer* [Le et al. 2012]. It is written in Java and includes the ability to synchronize speech, gestures, gaze, and facial animations. It integrates with dynamic listening behaviors and allows users to tweak and create their own gestures with a custom tool, which can be modulated for intensity. *SmartBody* [Shapiro 2011] ships as part of the VHToolkit, but is also a stand-alone charac-

[36] https://facefx.com

[37] https://blender.community/c/today/9sdbbc

[38] https://www.facewaretech.com

[39] https://docs.unrealengine.com/en-US/Engine/Animation/FacialRecordingiPhone/index.html

ter simulation platform available at https://smartbody.ict.usc.edu under the LGPL license. It is written in C++ and includes its own renderer and debugging tools, offers multiplatform support, and includes speech, locomotion, gazing, object manipulation, and physical simulation. *AsapRealizer 2.0* is part of Asap, with a particular focus on incremental behavior plan construction, graceful interruption, and adaptation of ongoing behavior. More information can be found in [Van Welbergen et al. 2014], which also includes a detailed comparison to other BML realizers. AsapRealizer is developed in Java and is available at https://github.com/ArticulatedSocialAgentsPlatform/AsapRealizer under the LGPL v3 license. *LiteBody* is part of the overall Relational Agents approach [Bickmore et al. 2009]. LiteBody is specifically developed for the web, however, it requires Flash, which is no longer supported. It is developed in Java and available through https://relationalagents.com/demo/litebody.

20.6 Tools to Model Interactions

In this section we discuss tools that address human–SIA interaction. In particular: *speech recognition*, which turns user speech into text; *audio-visual sensing*, which perceives and analyzes the user's face, body and voice; *natural language processing*, which understands the user's verbal input, generates the character's verbal output, and manages the overall dialogue; and *expressive speech*, which generates character speech based on its communicative intent.

20.6.1 Speech Recognition

Speech recognition turns user speech into text the rest of the system can use. Systems process user audio either locally on the device or on a server in the cloud. Local systems are typically more secure, don't require an Internet connection, and are more flexible, allowing, for instance, the definition of custom acoustic or language models. Acoustic models describe the acoustic qualities of a target audience (e.g., you, you in a specific recording environment, accents, children vs. adults). Language models describe the linguistic qualities of a target domain (e.g., specialized vocabulary, common phrases). Cloud-based services have access to more computing power and therefore can have a higher accuracy, while typically costing money and requiring an Internet connection which can add latency. Either approach can provide sequential results (i.e., audio is processed once the user stops talking) or continuous results (i.e., audio is continuously processed, and text strings are intermittently sent). Most solutions provide a confidence score for the recognition and may provide additional information beyond the text (e.g., prosody, emotion, filler word removal).

One of the original local solutions is the *CMUSphinx* suite [Lamere et al. 2003]. It contains several tools for both Java and C. It supports Windows, MacOS, Linux, and Android, and it allows you to create custom acoustic and language models. It is available at https://cmusphinx.github.io. While development on the core Sphinx suite has slowed down, it provides links to related efforts. *Kaldi* is a research-focused, local speech recognition toolkit for Windows, MacOS, Unix/Linux, and Android [Povey et al. 2011]. It is written in C++ and has an active

development community. It is available under the Apache 2 license at https://kaldi-asr.org. For feature extraction (e.g., pitch, voice activation detection), *OpenSmile* [Eyben et al. 2013] at https://www.audeering.com/opensmile and *PRAAT* at https://www.praat.org are popular tools.

Cloud solutions are typically provided by large technology companies, driven by voice and personal assistant applications. This has resulted in big improvements in speed and accuracy, which can be leveraged for SIA. The main ones include Google Speech-to-Text,[40] Amazon Transcribe,[41] Microsoft Azure Speech to Text,[42] and IBM Speech to Text.[43] These services are able to leverage large quantities of data and computing power in order to provide solid accuracy and a single solution for multiple hardware platforms, at the cost of customization, possible data collection, and service fees.

Most hardware and Operating System (OS) platforms also offer APIs for native speech recognition solutions, for example Microsoft Windows,[44] Apple MacOS and iOS,[45] and Android Speech.[46]

20.6.2 Audio-visual Sensing

Audio-visual sensing uses cameras and microphones to perceive a user's face, body, or voice, in order to recognize facial features, gestures, voice acoustics, and so on. This can be used for a variety of purposes, for example to recognize people, an increasingly controversial use. Within the context of SIA, audio-visual sensing is typically used to detect the affect of the user, in particular in support of real-time conversational goals, including rapport building and improved understanding through nonverbal behavior. As with speech recognition, tools are either local or cloud-based.

Local solutions include *OpenCV* (Open Source Computer Vision Library), an open source computer vision and machine learning software library that aims to provide a common infrastructure for a range of computer vision-related applications, including detecting faces and classifying human actions. It is written in C++ and supports Windows, Linux, Mac OS, and Android. It is freely available at https://github.com/opencv/opencv under the BSD license.

Social Signal Interpretation (SSI) is a framework for real-time recognition of social signals, including tools to record, analyze and recognize human behavior in real-time, such as gestures, mimics, head nods, and emotional speech [Wagner et al. 2013]. SSI allows the integration of multiple data producers and consumers for both audio and video. It is written in C++ and available under the GPL v3 and LGPL v3 licenses at https://hcm-lab.de/projects/ssi.

[40] https://cloud.google.com/speech-to-text

[41] https://aws.amazon.com/transcribe

[42] https://azure.microsoft.com/en-us/services/cognitive-services/speech-to-text

[43] https://www.ibm.com/topics/speech-recognition

[44] https://docs.microsoft.com/en-us/windows/apps/speech

[45] https://developer.apple.com/documentation/speech

[46] https://developer.android.com/reference/android/speech/package-summary

Closely related to SSI is the *NOnVerbal behaviour Analysis* tool (NovA) [Baur et al. 2013], which supports the analysis and interpretation of social signals conveyed by gestures, facial expressions and others as a basis for computer-enhanced social coaching. See https://github.com/hcmlab/nova for more details.

MultiSense [Stratou and Morency 2017] is part of the VHToolkit (see section 20.4.3.2). MultiSense combines multiple existing and custom data producers and consumers into a single framework, based on SSI, see below. It primarily focuses on the face using commodity web cams, but can analyze the full body with the original Microsoft Kinect and has the possibility to add custom voice analytics modules. Results are communicated through message passing using PML. See [Scherer et al. 2012] for more details.

OpenSMILE, despite its name, is focused mainly on acoustic analysis of voice and music. SMILE stands for Speech and Music Interpretation by Large-space Extraction and can be used in both real-time as well as offline feature extraction on large datasets. Within the context of SIA it is most useful for voice activation detection and speech emotion recognition. OpenSMILE works on Windows, Linux, and Mac OS. Its source code is available under a custom license at https://www.audeering.com/opensmile.

OpenFace 2.0 performs real-time facial landmark detection, head pose estimation, facial action unit recognition, and eye-gaze estimation using a webcam [Baltrusaitis et al. 2018]. The underlying models can be re-trained and the source code is freely available for research purposes at https://github.com/TadasBaltrusaitis/OpenFace.

OpenPose offers 2D pose detection, including body, foot, hand, and facial landmarks [Cao et al. 2018]. It can detect these in real-time for multiple people in an image or video stream. OpenPose comes with a C++ and Python API. The source code is available for non-commercial use at https://github.com/CMU-Perceptual-Computing-Lab/openpose.

Microsoft Platform for Situated Intelligence (\psi) is an open source framework for multimodal intelligent systems [Bohus et al. 2017]. It consists of a runtime for real-time data collection and manipulation, a suite of tools for analytics, visualization and training, and a collection of components that can be combined to create applications. It uses C# with interfaces to other languages (e.g., Python, JavaScript). It integrates with Azure Cognitive Services, as well as the Azure Kinect DK (https://azure.microsoft.com/en-us/services/kinect-dk). \psi is available at https://github.com/microsoft/psi under the MIT license.

OpenSense follows MultiSense and provides a framework in which producers and consumers can be combined in a flexible way [Stefanov et al. 2020]. It builds on \psi (see below) and is written mainly in C# and C++. It is available at https://github.com/intelligent-human-perception-laboratory.

The big US tech companies offer web services for audio-visual sensing as well. The main focus is visual, including face detection, facial landmark detection, emotion detection, and object recognition. These services can be accessed directly through REST calls or using

dedicated SDKs for the most common languages. Services include Amazon Rekognition,[47] Google Vision,[48] and Microsoft Azure Computer Vision.[49]

20.6.3 Natural Language Processing

Natural language processing (NLP) can be divided into three areas:

- Natural language understanding: comprehend what the user is saying.
- Natural language generation: generate what the agent should say.
- Dialogue management: manage the conversation between two or more entities.

Tools can cover a mix of these three areas. One important element is who has the initiative in the conversation: the user (e.g., a question-answering system, personal assistant), the agent (e.g., virtual interviewer), or both. The latter, mixed-initiative systems, can handle more complex conversations and are typically more difficult to develop. For a more in-depth discussion of natural language approaches, see Chapter 5 on "Natural Language Understanding in Socially Interactive Agents" [Pieraccini 2021] of volume 1 of this handbook [Lugrin et al. 2021]. Here, we will primarily discuss common tools that can be used in real-time systems. Given the focus on digital and voice assistants, this is a very active area of research and development, performed at both academia and within industry. This results in many available tools ranging from individual libraries for natural language research to fully developed solutions and everything in between, often made open source, regardless of the origin.

Olympus, from Carnegie Melon University (CMU), is one of the earliest available NLP tools. It is a suite of tools that cover speech recognition, natural language understanding and generation, and dialogue management [Bohus et al. 2007]. *RavenClaw* is the dialogue manager and can handle mixed-initiative conversations that match external input to an internal agenda linked to a task model [Bohus and Rudnicky 2009]. The tools in Olympus are available under the BSD license at http://wiki.speech.cs.cmu.edu/olympus.

The *NPCEditor* is part of the VHToolkit (see Section 20.4.3.2). It is a statistical text classifier that matches novel user input to the best pre-authored character response output using an information retrieval approach [Leuski and Traum 2011]. Authors provide the NPCEditor with examples of how user input should be matched to character output. Novel user input is analyzed against close known inputs and their linked outputs, resulting in a set of possible answers above a certain threshold. This set is processed by its dialogue manager, a Groovy script that can be customized. For example, it can avoid repeating the same utterance or prompt the user if no suitable response can be retrieved.

[47] https://aws.amazon.com/rekognition

[48] https://cloud.google.com/vision

[49] https://azure.microsoft.com/en-us/services/cognitive-services/computer-vision

OpenDial is a toolkit with a focus on dialogue management, with the possibility to add natural language understanding and generation, text-to-speech, and multimodal processing [Lison and Kennington 2016]. It is written in Java and provides a hybrid approach that combines human readable rules with a Bayesian network that contains the dialogue state. It is available through http://www.opendial-toolkit.net under the MIT license and seems to no longer be in active development.

PyDial, the Cambridge University Python Multi-domain Statistical Dialogue System Toolkit, is a more research-focused toolkit [Ultes et al. 2017]. It offers natural language understanding and generation, as well as dialogue management, provided by both rule-based and model-based approaches. PyDial uses Python and is available at http://www.camdial.org/pydial under the Apache 2.0 license.

Rasa is a commercial company that offers an open source solution for conversational AI, including dialogue management and natural understanding [Bocklisch et al. 2017]. More advanced services, including annotations, multiple deployed versions, and support require a subscription. Rasa is written in Python and the open source portion is available under the Apache 2.0 license through https://rasa.com.

DeepPavlov is an open source library for creating natural language solutions, combining machine learning and deep leaning models with traditional rule-based approaches. These form the basis for individual skills (e.g., question-answering, goal-oriented dialogue) that can be combined into a single agent, which can be integrated with existing systems of services. DeepPavlov uses TensorFlow and Keras,[50] supports Windows and Linux, and mainly uses Python. It is available at https://deeppavlov.ai under the Apache 2.0 license.

ChatScript is a rule-based scripting language that forms the foundation for many custom natural language systems (see https://github.com/ChatScript/ChatScript). It does this through describing patterns of user input, combined with an ontology and built-in memory of conversations. ChatScript works on Windows, Linux, MacOS, iOS, and Android, and has a server version. It is available under the MIT license.

The main big US tech companies offer a suite of natural language processing services, including text understanding, semantic analysis, and conversational interactions, for example, Google Dialogfow,[51] Amazon Lex,[52] and Microsoft LUIS.[53] These are typically focused on personal assistant type interactions that drive Amazon Echo, Google Assistant, and Microsoft Cortana. This means conversations are authored around user intents (e.g., play music, book a vacation), with parameters to be filled in for the specifics of each request. Online authoring tools are aimed at domain experts rather than natural language researchers. The ability to connect to a service from any device offers flexibility, at the cost of requiring an online

[50] https://keras.io

[51] https://dialogflow.com

[52] https://aws.amazon.com/lex

[53] https://www.luis.ai

connection, providing user data, and paying for used data and compute cycles. These and other large companies increasingly open source parts of their technology stack, including Pluto from Uber,[54] ParlAI from Facebook,[55] and Google BERT[56] and ALBERT.[57] They join forces with more traditional academic approaches, including Stanford's suite of software.[58]

20.6.4 Expressive Speech

Expressive speech, or text-to-speech generation, turns character text into speech (see Chapter 6 on "Building and Designing Expressive Speech Synthesis" [Aylett et al. 2021] of volume 1 of this handbook [Lugrin et al. 2021]). Most tools allow at a minimum the definition of a particular voice and optionally allow annotations of the speech with SSML [Taylor and Isard 1997] to, for instance, indicate emphasis, prosody or emotion. In order to utilize these tools in real-time, in addition to the resulting audio file, they need to provide a phoneme schedule, in order to drive lip-synching, see Section 20.5.2.

One of the earliest available tools is *Festival* [Taylor et al. 1998]. This is an offline available tool that offers various male and female voices, using a range of approaches. While some of these approaches are outdated, it provides an out-of-the-box solution that is relatively easy to integrate, using C++. It supports English and Spanish and is available at http://www.cstr.ed.ac.uk/projects/festival.

MaryTTS supports roughly ten languages with a host of options for metadata, including phoneme and intonation schedules using a custom XML schema [Schröder et al. 2011]. It is written in Java and can be used locally or set up as a service. MaryTTS is available at http://mary.dfki.de under the LGPL v3 license.

There are several commercial options available. *CereVoice*[59] was one of the first viable companies in this space and offers both local and online solutions, with either a proprietary SDK or web services [Aylett and Pidcock 2007]. Just as with other personal and voice assistant-related technologies, the big tech companies offer online services, including from Amazon,[60] Google,[61] and Microsoft.[62] Many of these offer the functionality to clone a voice as well, where a relatively small amount of voice data from a specific person can be used to generate novel speech by that same person [Arik et al. 2018]. This allows pre-recorded speech to be matched with generated speech. Finally, much work is currently put into making generated voices sound more natural, including introducing disfluencies [Oord et al. 2016].

[54] https://github.com/uber-research/plato-research-dialogue-system

[55] https://github.com/facebookresearch/ParlAI

[56] https://github.com/google-research/bert

[57] https://github.com/google-research/ALBERT

[58] https://nlp.stanford.edu/software

[59] https://www.cereproc.com

[60] https://aws.amazon.com/polly

[61] https://cloud.google.com/text-to-speech

[62] https://azure.microsoft.com/en-us/services/cognitive-services/text-to-speech

20.7 Similarities and Differences in IVAs and SRs

Both IVAs and SRs aim to perceive a human, process that input and integrate it with its internal state in order to respond appropriately both verbally and nonverbally. However, while IVAs inhabit a virtual world, SRs are fully embedded within the real world, which brings with it many additional challenges and constraints, including perception (e.g., mapping the real, dynamic world to an internal representation), physics (e.g., using mechanical elements to realize conversational gestures), navigation (combining perception and physics), and energy (balancing the power of an energy source with volume, weight and mobility). One school of thought views this physical embodiment and being part of the real world as a fundamental necessity of creating SIAs. Per [Brooks 1991], human evolution took a long time to create low-level systems to interact with the world, on top of which intelligence evolved relatively quickly. In addition, the complexity of a system itself may be determined by the complexity of the environment it operates in.[63] Finally, by abstracting the real world, researchers and developers necessarily do the heavy intellectual lifting in these abstractions rather than in the systems themselves and may create simplifications and dependencies that cannot be overcome when these systems are deployed in the real world.

Many of the platforms and tools discussed here can be and have been applied to robotics. For instance, Greta has been integrated with an Aibo robot[64] [Niewiadomski et al. 2009] and a Nao[65] robot [Le and Pelachaud 2011]. The VHToolkit has also been used with a Nao robot [Artstein et al. 2016] as well as with mobile robots that use a traditional monitor screen to display a virtual head on [Pang et al. 2018, Si and McDaniel 2016], while SmartBody is used in the Sophia robot by Hanson[66].

Mixed approaches like these are more commonplace, aiming to combine advantages from IVAs with SRs. *Furhat* takes a novel approach in back projecting a virtual face on a physical mold in order to address the limitations of a 2D screen, in particular in regard to gaze behavior [Al Moubayed et al. 2012]. It allows for multiparty interactions in a 3D physical space while leveraging smooth facial expressions. Furhat is driven by *IrisTK*, which is a modular, Java-based system with the specific aim of supporting SR research and development [Skantze and Al Moubayed 2012]. IrisTK is available at http://www.iristk.net under the GPL v3 license.

Another approach is to leverage the virtual space to simulate robots before physically creating them in order to speed up research and development [Coevoet et al. 2017]. For instance, work with the VHToolkit has shown that new algorithms can be rapidly iterated upon in a simulation, where physical and time constraints are reduced, before implementing promising candidates in the real world. This includes experimentation with a larger number

[63] For example, the complex route an ant takes to get home is largely a reflection of the obstacles and entities it encounters in its environment rather than of its quite simple set of behaviors [Simon 1969].

[64] https://us.aibo.com

[65] https://www.softbankrobotics.com/emea/en/nao

[66] https://www.hansonrobotics.com/press_release/hanson-robotics-limited-partners-with-embody-digital

of robots than may be feasible in the real world, more advanced sensors than are currently available, or multiple labs in different locations [Hönig et al. 2015].

Similarly, shared frameworks allow for the exploration of IVA and SR communication [Rahman 2019]. *ScoutBot* is developed using the VHToolkit and is integrated with the *Robot Operating System* (ROS) [Lukin et al. 2018]. ROS[67] is an open collection of tools, libraries, and conventions that support a wide range of robot research and development [Quigley et al. 2009]. Dedicated migration platforms allow for an agent to migrate between different embodiments (e.g., from an IVA to an SR) [Hassani and Lee 2014, Kriegel et al. 2011].

While IVA research typically focuses on simulating human appearance and behavior, this may not be desirable for SRs [Gratch et al. 2015]. Abstracting away from realistic humans to animals or stylized humanoid robots allows for focusing on the social and interactive elements without the challenge of re-creating lifelike, physical human representation. Examples include Aibo and Nao mentioned above, the Jibo robot[68], and the Nabaztag robot[69], the last of which has been supported by Asap Realizer [Reidsma and van Welbergen 2013]. Recently, Embodied revealed their robot, Moxie, a lifelike robot-companion for children to provide support in the development of social and emotional skills, as described in their recent study [Hurst et al. 2020].

Regardless of the type of robot, most benefit from the tools discussed in this chapter, in particular audio-visual sensing, speech recognition, natural language processing, and expressive speech. More humanoid-like characters also benefit from nonverbal behavior generation [Matsuyama 2015, Mead et al. 2010, Mlakar et al. 2013]). With ever more advanced capabilities and tools, increased fidelity in graphics, and improved hardware, we're bound to see the cross-pollination between IVAs and SRs that has started at the beginning of the field continue to increase in the coming years.

20.8 Current Challenges

While much progress has been made in the past 20 to 30 years in regard to platforms and tools, as well as their effectiveness and dissemination, many challenges remain.

In individual specializations, these are often directly tied to the challenges of the field itself. Modeling the human mind, for example, or accurately inferring a person's mental state based on audio-visual input are by no means solved problems. As a result, theories, approaches, and technologies have not yet matured to a point where they can be elegantly captured in user-friendly solutions for further use in research or development.

However, much progress has been made in individual fields driven by the rise of machine learning techniques. This approach does bring its own set of challenges. Effective machine learning requires huge amounts of data, which is difficult for smaller teams to acquire. Even

[67] https://www.ros.org

[68] https://jibo.com

[69] http://www.freerabbits.nl

for larger teams, properly curating datasets is not only labor-intensive, but it is rife with challenges regarding biases, which is only exacerbated by the inherent black-box nature of most machine learning techniques. Effective tools will require validated datasets combined with explainable AI features in order to create systems that can be trusted to give the intended results.

As a whole, the combined trends of increased specialization and expanded democratization of many SIA relevant technologies, partly driven by the push of digital personal assistants, has led to an explosion of available tools. This increases the potential of being able to leverage existing capabilities while also increasing its complexity. Furthermore, relatively little progress has been made to standardize interfaces in such a way that individual requirements can be met by interchangeable solutions. This is due to many factors, including diverse sets of requirements (e.g., voice-based call center interactions vs. embodied story-driven agents), divergent incentives (e.g., research vs. commercialization), the availability of multiple hardware platforms and form factors (e.g., web, mobile, desktop, AR, VR), and the fact that formalizing principled representations on how exactly the human mind and body operate is just, well, hard. However, the lack of formal standards does not prevent software as a whole to become ever more modular, allowing distributed systems to be developed more organically from available microservices. This allows researchers and developers to pick and choose from an ever-expanding suite of relevant services at the cost of interfacing with them individually.

This points to one of the most enduring remaining challenges: providing an integrated solution that provides the tools to create and validate both appearance and behaviors in all their nuanced interplays. This requires a deep level of understanding of not only the individual research fields, but also how they all interconnect; a level we have not yet reached, neither within the social sciences nor the "hard" sciences.

Even while we continue to gain in our level of understanding, creating solid tools that are easy to use is a challenge in and of itself. Developing these tools require (1) a deep understanding of all underlying research fields that the tool aims to capture, (2) decisions regarding the trade-off between power and complexity as well as the possible abstraction levels for each, and (3) a solid understanding of the end users, their skill level, their likely knowledge of the domain, and how they can and want to leverage the tool in a user-friendly manner. As with the research that underlies all aspects of SIA, this requires an interdisciplinary team of researchers and developers, with the additional challenge of translating gained knowledge, capabilities, and procedures to a user-friendly package that the end user can take advantage of.

20.9 Future Directions

As our understanding of a particular SIA field advances and the maturity of related technologies increases, the tools that support these grow increasingly more powerful and user-friendly. The current interest in SIA in general and the associated commercial applications in particular will lead to a continuous democratization of tools that support SIA exploration and creation.

As before, this starts with relatively isolated aspects of human behavior (e.g., speech, hearing) and continues with more complex behavior (e.g., multiparty dialogue, longitudinal relationships). As these tools become more advanced and user-friendly, it lowers the barrier of entry to the inherently integrated nature of SIA research and development.

In terms of platforms and tools, the authors aim to focus on the following aspects:

1. *Microservices architecture*. As per [Hartholt et al. 2020], the aim is to provide a modern, modular architecture that leverages cloud services in order to offer both researchers and developers a powerful and flexible framework to collaboratively explore SIAs. The modularity allows for multiple implementations for a given service, allowing for tradeoffs between power, flexibility, and performance.
2. *Multiplatform support*. The aim is to enhance the VHToolkit to not only support desktop applications, but also web, mobile, AR and VR, in order to better explore the strengths of each platform [Hartholt et al. 2019a]. This leverages the microservices architecture and provides per-platform capabilities and best practices.
3. *Audio-visual sensing*. The more real-time information can be gathered from the user, the better SIAs are positioned to converse with end users in a manner that increases rapport and avoids frustration. This requires more research to go from external feature extraction (e.g., smile, frown) to internal inference (e.g., happy, confused), for all members of the human population. The aim is to integrate a range of cloudservices and local solutions (e.g., [Stefanov et al. 2020]) to provide a testbed for integrated exploration.
4. *Character generation*. As exploring our own humanity is a key pillar of SIA research, we collectively should strive for IVAs to match our diversity in order to (1) represent populations from all over the world, (2) support social research related to race, ethnicity, gender, sexuality, age, and so on, and (3) avoid unnatural repetition. It is therefore vital that we lower the required effort to create diverse, high-quality IVAs. We aim to pursue this by leveraging recent progress in generating high-fidelity characters [Li et al. 2020].
5. *Integrated authoring tools*. Humans are complex beings with advanced capabilities that have no clear boundaries; it is the system as a whole that leads to complex behaviors and overall intelligence. Similarly, SIAs require tightly integrated capabilities that collectively realize the goals for which they have been created. This requires authoring tools that take the integrated nature of SIAs into account and provide ways to create and validate agents that encompasses the whole rather than solely focus on individual areas.

20.10 Summary

One of the defining aspects of humanity is the use of tools to increase our productivity and enhance our understanding of ourselves and the world we live in. As our tools have become more sophisticated, so has our ability to create and understand. There is no better field

than SIA to exemplify this duality of creation and understanding, from exploring ourselves mentally, physiologically, and socially, to developing theories and implementations of virtual counterparts. It is tools that underlie this intertwined process and allow us to progress.

The sophistication of SIAs have advanced considerably in recent years, and with it, more and more tools have been made available to support the research and development of both IVAs and SRs. As a result, the barriers to enter this important field have been lowered to the point where it is easier than ever for individuals and small groups to advance our understanding of what makes us human and to leverage that knowledge in creating applications that benefit humanity.

Big challenges remain, given the complexity of the subject matter. Progress in individual areas will advance our understanding and lead to ever more powerful tools. These will have to come together in order to provide a holistic approach to researching and developing SIAs. This requires an interdisciplinary approach, where researchers, developers, artists, and usability experts work together to understand, create, and refine the tools that enable not only the creation of powerful new systems, but the understanding of the human body and mind itself.

Bibliography

J. Achenbach, T. Waltemate, M. E. Latoschik, and M. Botsch. 2017. Fast generation of realistic virtual humans. *Proceedings of the ACM Symposium on Virtual Reality Software and Technology, VRST*, Part F131944. DOI: 10.1145/3139131.3139154.

S. Al Moubayed, J. Beskow, G. Skantze, and B. Granström. 2012. Furhat: a back-projected human-like robot head for multiparty human-machine interaction. In *Cognitive behavioural systems*, pp. 114–130. Springer.

T. Alldieck, G. Pons-Moll, C. Theobalt, and M. Magnor. 2019. Tex2shape: Detailed full human body geometry from a single image. In *IEEE International Conference on Computer Vision (ICCV)*. IEEE.

J. R. Anderson, M. Matessa, and C. Lebiere. 1997. Act-r: A theory of higher level cognition and its relation to visual attention. *Human–Computer Interaction*, 12(4): 439–462.

K. Anderson, E. André, T. Baur, S. Bernardini, M. Chollet, E. Chryssafidou, I. Damian, C. Ennis, A. Egges, P. Gebhard, et al. 2013. The tardis framework: intelligent virtual agents for social coaching in job interviews. In *International Conference on Advances in Computer Entertainment Technology*, pp. 476–491. Springer.

E. André, T. Rist, and J. Müller. 1998. Webpersona: a lifelike presentation agent for the world-wide web. *Knowledge-Based Systems*, 11(1): 25–36.

Y. Arafa and A. Mamdani. 2003. Scripting embodied agents behaviour with cml: character markup language. In *Proceedings of the 8th international conference on Intelligent user interfaces*, pp. 313–316.

S. Arik, J. Chen, K. Peng, W. Ping, and Y. Zhou. 2018. Neural voice cloning with a few samples. In *Advances in Neural Information Processing Systems*, pp. 10019–10029.

R. Artstein, S. Gandhe, A. Leuski, and D. Traum. 2008. Field testing of an interactive question-answering character. In *ELRA Workshop on Evaluation Looking into the Future of Evaluation: When automatic metrics meet task-based and performance-based approaches*, p. 36.

R. Artstein, D. Traum, J. Boberg, A. Gainer, J. Gratch, E. Johnson, A. Leuski, and M. Nakano. 2016. Niki and julie: a robot and virtual human for studying multimodal social interaction. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 402–403.

M. P. Aylett and C. J. Pidcock. 2007. The cerevoice characterful speech synthesiser sdk. In *IVA*, pp. 413–414.

M. P. Aylett, L. Clark, B. R. Cowan, and I. Torre. 2021. *The Handbook on Socially Interactive Agents: 20 years of Research on Embodied Conversational Agents, Intelligent Virtual Agents, and Social Robotics Volume 1: Methods, Behavior, Cognition*, pp. 173–211. ACM Press, 1.

T. Baltrusaitis, A. Zadeh, Y. C. Lim, and L.-P. Morency. 2018. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pp. 59–66. IEEE.

T. Baur, I. Damian, F. Lingenfelser, J. Wagner, and E. André. 2013. Nova: Automated analysis of nonverbal signals in social interactions. In *International Workshop on Human Behavior Understanding*, pp. 160–171. Springer.

C. Becker-Asano and I. Wachsmuth. 2010. Affective computing with primary and secondary emotions in a virtual human. *Autonomous Agents and Multi-Agent Systems*, 20(1): 32.

F. Bergmann and B. Fenton. 2015. Scene based reasoning. In *International Conference on Artificial General Intelligence*, pp. 25–34. Springer.

E. Bevacqua, S. Pammi, S. J. Hyniewska, M. Schröder, and C. Pelachaud. 2010. Multimodal backchannels for embodied conversational agents. In *International Conference on Intelligent Virtual Agents*, pp. 194–200. Springer.

T. Bickmore, D. Schulman, and G. Shaw. 2009. Dtask and litebody: Open source, standards-based tools for building web-deployed embodied conversational agents. In *International Workshop on Intelligent Virtual Agents*, pp. 425–431. Springer.

T. Bickmore, D. Schulman, and L. Yin. 2010. Maintaining engagement in long-term interventions with relational agents. *Applied Artificial Intelligence*, 24(6): 648–666.

T. W. Bickmore, D. Schulman, and C. L. Sidner. 2011. A reusable framework for health counseling dialogue systems based on a behavioral medicine ontology. *Journal of biomedical informatics*, 44(2): 183–197.

A. Black, P. Taylor, R. Caley, and R. Clark, 1998. The festival speech synthesis system.

T. Bocklisch, J. Faulkner, N. Pawlowski, and A. Nichol. 2017. Rasa: Open source language understanding and dialogue management. *arXiv preprint arXiv:1712.05181*.

D. Bohus and A. I. Rudnicky. 2009. The ravenclaw dialog management framework: Architecture and systems. *Computer Speech & Language*, 23(3): 332–361.

D. Bohus, A. Raux, T. Harris, M. Eskenazi, and A. Rudnicky. 2007. Olympus: an open-source framework for conversational spoken language interface research. In *Proceedings of the workshop on bridging the gap: Academic and industrial research in dialog technologies*, pp. 32–39.

D. Bohus, S. Andrist, and M. Jalobeanu. 2017. Rapid development of multimodal interactive systems: a demonstration of platform for situated intelligence. In *Proceedings of the 19th ACM International Conference on Multimodal Interaction*, pp. 493–494.

R. A. Brooks. 1991. Intelligence without representation. *Artificial intelligence*, 47(1-3): 139–159.

S. L. Burke, T. Bresnahan, T. Li, K. Epnere, A. Rizzo, M. Partin, R. M. Ahlness, and M. Trimmer. 2018. Using virtual interactive training agents (vita) with adults with autism and other developmental disabilities. *Journal of autism and developmental disorders*, 48(3): 905–912.

Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. 2018. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *arXiv preprint arXiv:1812.08008*.

J. Cassell, C. Pelachaud, N. Badler, M. Steedman, B. Achorn, T. Becket, B. Douville, S. Prevost, and M. Stone. 1994. Animated conversation: Rule-based generation of facial expression, gesture & spoken intonation for multiple conversational agents. *Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1994*, pp. 413–420. DOI: 10.1145/192161.192272.

J. Cassell, J. Sullivan, S. Prevost, and E. Churchill. 2000. Embodied Conversational Agents edited by. Technical report.

J. Cassell, H. H. Vilhjálmsson, and T. Bickmore. 2001. BEAT: The Behavior Expression Animation Toolkit. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 2001*, pp. 477–486. Association for Computing Machinery. ISBN 158113374X. DOI: 10.1145/383259.383315.

J. Cassell, H. H. Vilhjálmsson, and T. Bickmore. 2004. Beat: the behavior expression animation toolkit. In *Life-Like Characters*, pp. 163–185. Springer.

J. Chibane, T. Alldieck, and G. Pons-Moll. jun 2020. Implicit functions in feature space for 3d shape reconstruction and completion. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE.

E. Coevoet, T. Morales-Bieze, F. Largilliere, Z. Zhang, M. Thieffry, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, J. Dequidt, et al. 2017. Software toolkit for modeling, simulation, and control of soft robots. *Advanced Robotics*, 31(22): 1208–1224.

D. Crevier. 1993. *AI: the tumultuous history of the search for artificial intelligence*. Basic Books, Inc.

B. De Carolis, C. Pelachaud, I. Poggi, and F. de Rosis. 2001. Behavior planning for a reflexive agent. In *International Joint Conference on Artificial Intelligence*, volume 17, pp. 1059–1066. LAWRENCE ERLBAUM ASSOCIATES LTD.

B. De Carolis, V. Carofiglio, and C. Pelachaud. 2002. From discourse plans to believable behavior generation. In *Proceedings of the International Natural Language Generation Conference*, pp. 65–72.

P. Debevec, T. Hawkins, C. Tchou, H.-P. Duiker, W. Sarokin, and M. Sagar. 2000. Acquiring the reflectance field of a human face. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 145–156.

E. Douglas-Cowie, C. Cox, J.-C. Martin, L. Devillers, R. Cowie, I. Sneddon, M. McRorie, C. Pelachaud, C. Peters, O. Lowry, et al. 2011. The humaine database. In *Emotion-Oriented Systems*, pp. 243–284. Springer.

I. Doumanis. 2013. *Evaluating humanoid embodied conversational agents in mobile guide applications*. PhD thesis, Middlesex University.

R. Ekman. 1997. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. Oxford University Press, USA.

M. Evers and A. Nijholt. 2000. Jacob-an animated instruction agent in virtual reality. In *International Conference on Multimodal Interfaces*, pp. 526–533. Springer.

F. Eyben, F. Weninger, F. Gross, and B. Schuller. 2013. Recent developments in opensmile, the munich open-source multimedia feature extractor. In *Proceedings of the 21st ACM international conference on Multimedia*, pp. 835–838.

A. W. Feng, A. Leuski, S. Marsella, D. Casas, S.-H. Kang, and A. Shapiro. 2015. A platform for building mobile virtual humans. In *International Conference on Intelligent Virtual Agents*, pp. 310–319. Springer.

T. Finin, R. Fritzson, D. McKay, and R. McEntire. 1994. Kqml as an agent communication language. In *Proceedings of the third international conference on Information and knowledge management*, pp.

456–463.

P. Gebhard, G. Mehlmann, and M. Kipp. 2012. Visual scenemaker—a tool for authoring interactive virtual characters. *Journal on Multimodal User Interfaces*, 6(1-2): 3–11.

J. Gratch, J. Rickel, E. André, J. Cassell, E. Petajan, and N. Badler. 2002. Creating interactive virtual humans: Some assembly required. *IEEE Intelligent systems*, 17(4): 54–63.

J. Gratch, S. Hill, L.-P. Morency, D. Pynadath, and D. Traum. 2015. Exploring the implications of virtual human research for human-robot teams. In *International Conference on Virtual, Augmented and Mixed Reality*, pp. 186–196. Springer.

D. Hart and B. Goertzel. 2008. Opencog: A software framework for integrative artificial general intelligence. In *AGI*, pp. 468–472.

A. Hartholt, J. Gratch, L. Weiss, et al. 2009. At the virtual frontier: Introducing gunslinger, a multi-character, mixed-reality, story-driven experience. In *International Workshop on Intelligent Virtual Agents*, pp. 500–501. Springer.

A. Hartholt, D. Traum, S. C. Marsella, A. Shapiro, G. Stratou, A. Leuski, L.-P. Morency, and J. Gratch. 2013. All together now, introducing the virtual human toolkit. In *International Workshop on Intelligent Virtual Agents*, pp. 368–381. Springer.

A. Hartholt, E. Fast, A. Reilly, W. Whitcup, M. Liewer, and S. Mozgai. 2019a. Ubiquitous virtual humans: A multi-platform framework for embodied ai agents in xr. In *2019 IEEE International Conference on Artificial Intelligence and Virtual Reality (AIVR)*, pp. 308–3084. IEEE.

A. Hartholt, S. Mozgai, E. Fast, M. Liewer, A. Reilly, W. Whitcup, and A. S. Rizzo. 2019b. Virtual humans in augmented reality: A first step towards real-world embedded virtual roleplayers. In *Proceedings of the 7th International Conference on Human-Agent Interaction*, pp. 205–207.

A. Hartholt, S. Mozgai, and A. S. Rizzo. 2019c. Virtual job interviewing practice for high-anxiety populations. In *Proceedings of the 19th ACM International Conference on Intelligent Virtual Agents*, pp. 238–240.

A. Hartholt, E. Fast, A. Reilly, W. Whitcup, M. Liewer, and S. Mozgai. 2020. Multi-platform expansion of the virtual human toolkit: Ubiquitous conversational agents. *International Journal of Semantic Computing*, 14(3).

K. Hassani and W.-S. Lee. 2014. On designing migrating agents: from autonomous virtual agents to intelligent robotic systems. In *SIGGRAPH Asia 2014 Autonomous Virtual Humans and Social Robot for Telepresence*, pp. 1–10.

M. J. Hays, J. C. Campbell, M. A. Trimmer, J. C. Poore, A. K. Webb, and T. K. King. 2012. Can role-play with virtual humans teach interpersonal skills? Technical report, UNIVERSITY OF SOUTHERN CALIFORNIA LOS ANGELES INST FOR CREATIVE TECHNOLOGIES.

J. Hendler. 2008. Avoiding another ai winter. *IEEE Intelligent Systems*, (2): 2–4.

D. Heylen, S. Kopp, S. Marsella, C. Pelachaud, and H. Vilhjalmsson. 2008a. Why conversational agents do what they do? functional representations for generating conversational agent behavior. In *The First Functional Markup Language Workshop. Estoril, Portugal*.

D. Heylen, S. Kopp, S. C. Marsella, C. Pelachaud, and H. Vilhjálmsson. 2008b. The next step towards a function markup language. In *International Workshop on Intelligent Virtual Agents*, pp. 270–280. Springer.

T. Holz, A. G. Campbell, G. M. O'Hare, J. W. Stafford, A. Martin, and M. Dragone. 2011. Mira—mixed reality agents. *International journal of human-computer studies*, 69(4): 251–268.

J. Howison and K. Crowston. 2004. The perils and pitfalls of mining sourceforge. In *MSR*, pp. 7–11. IET.

D. Huggins-Daines, M. Kumar, A. Chan, A. W. Black, M. Ravishankar, and A. I. Rudnicky. 2006. Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices. In *2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings*, volume 1, pp. I–I. IEEE.

N. Hurst, C. Clabaugh, R. Baynes, J. Cohn, D. Mitroff, and S. Scherer. 2020. Social and emotional skills training with embodied moxie. *arXiv preprint arXiv:2004.12962*.

W. Hönig, C. Milanes, L. Scaria, T. Phan, M. Bolas, and N. Ayanian. 2015. Mixed reality for robotics. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5382–5387. IEEE.

E. Kalliamvakou, G. Gousios, K. Blincoe, L. Singer, D. M. German, and D. Damian. 2014. The promises and perils of mining github. In *Proceedings of the 11th working conference on mining software repositories*, pp. 92–101.

R. Klaassen, J. Hendrix, D. Reidsma, and H. J. op den Akker. 2012. Elckerlyc goes mobile enabling technology for ecas in mobile applications. *UBICOMM 2012*.

S. Kopp and T. Hassan. 2022. *The Handbook on Socially Interactive Agents: 20 years of Research on Embodied Conversational Agents, Intelligent Virtual Agents, and Social Robotics Volume 2: Interactivity, Platforms, Application*, pp. 77–111. ACM Press, 2.

S. Kopp, B. Krenn, S. Marsella, A. N. Marshall, C. Pelachaud, H. Pirker, K. R. Thórisson, and H. Vilhjálmsson. 2006. Towards a Common Framework for Multimodal Generation: The Behavior Markup Language. Technical report.

S. Kopp, H. van Welbergen, R. Yaghoubzadeh, and H. Buschmeier. 2014. An architecture for fluid real-time conversational agents: integrating incremental output generation and input processing. *Journal on Multimodal User Interfaces*, 8(1): 97–108.

A. Kranstedt, S. Kopp, and I. Wachsmuth. 2002. Murml: A multimodal utterance representation markup language for conversational agents. In *AAMAS'02 Workshop Embodied conversational agents-let's specify and evaluate them!*

M. Kriegel, R. Aylett, P. Cuba, M. Vala, and A. Paiva. 2011. Robots meet ivas: a mind-body interface for migrating artificial intelligent agents. In *International Workshop on Intelligent Virtual Agents*, pp. 282–295. Springer.

F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. 2001. Factor graphs and the sum-product algorithm. *IEEE Transactions on information theory*, 47(2): 498–519.

S. Kshirsagar, N. Magnenat-Thalmann, A. Guye-Vuillème, D. Thalmann, K. Kamyab, and E. Mamdani. 2002. Avatar markup language. In *ACM International Conference Proceeding Series*, volume 23, pp. 169–177.

J. E. Laird. 2012. *The Soar cognitive architecture*. MIT press.

J. E. Laird, K. R. Kinkade, S. Mohan, and J. Z. Xu. 2012. Cognitive robotics using the soar cognitive architecture. In *Workshops at the twenty-sixth AAAI conference on artificial intelligence*.

P. Lamere, P. Kwok, E. Gouvea, B. Raj, R. Singh, W. Walker, M. Warmuth, and P. Wolf. 2003. The cmu sphinx-4 speech recognition system. In *IEEE Intl. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2003), Hong Kong*, volume 1, pp. 2–5.

S. Larsson and D. Traum, 2000. Information state and dialogue management in the TRINDI Dialogue Move Engine Toolkit.

Q. Le, J. Huang, and C. Pelachaud. 2012. A common gesture and speech production framework for virtual and physical agents. In *ACM international conference on multimodal interaction*.

Q. A. Le and C. Pelachaud. 2011. Generating co-speech gestures for the humanoid robot nao through bml. In *International Gesture Workshop*, pp. 228–237. Springer.

J. Lee and S. Marsella. 2006. Nonverbal behavior generator for embodied conversational agents. In *International Workshop on Intelligent Virtual Agents*, pp. 243–255. Springer.

J. Lee, S. Marsella, D. Traum, J. Gratch, and B. Lance. 2007. The rickel gaze model: A window on the mind of a virtual human. In *International workshop on intelligent virtual agents*, pp. 296–303. Springer.

A. Leuski and D. Traum. 2011. Npceditor: Creating virtual human dialogue using information retrieval techniques. *Ai Magazine*, 32(2): 42–56.

R. Li, K. Bladin, Y. Zhao, C. Chinara, O. Ingraham, P. Xiang, X. Ren, P. Prasad, B. Kishore, J. Xing, et al. 2020. Learning formation of physically-based face attributes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3410–3419.

P. Lison and C. Kennington. 2016. Opendial: A toolkit for developing spoken dialogue systems with probabilistic rules. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Demonstrations)*, pp. 67–72. Association for Computational Linguistics, Berlin, Germany.

B. Lugrin, C. Pelachaud, and D. Traum. 2021. *The Handbook on Socially Interactive Agents: 20 years of Research on Embodied Conversational Agents, Intelligent Virtual Agents, and Social Robotics Volume 1: Methods, Behavior, Cognition*. ACM Press. DOI: https://doi.org/10.1145/3477322.

S. M. Lukin, F. Gervits, C. J. Hayes, A. Leuski, P. Moolchandani, J. G. Rogers III, C. S. Amaro, M. Marge, C. R. Voss, and D. Traum. 2018. Scoutbot: A dialogue system for collaborative navigation. *arXiv preprint arXiv:1807.08074*.

A. Marriott. 2001. Vhml–virtual human markup language. In *Talking Head Technology Workshop, at OzCHI Conference*, pp. 252–264.

R. Marvin, Jul 2018. How unity is building its future on ar, vr, and ai. https://www.pcmag.com/news/how-unity-is-building-its-future-on-ar-vr-and-ai.

Y. Matsuyama. 2015. *Multiparty Conversation Facilitation Robots*. PhD thesis, .

D. McClosky, E. Charniak, and M. Johnson. 2006. Reranking and self-training for parser adaptation. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pp. 337–344. Association for Computational Linguistics.

R. Mead, E. Wade, P. Johnson, A. S. Clair, S. Chen, and M. J. Matarić. 2010. An architecture for rehabilitation task practice in socially assistive human-robot interaction. In *19th International Symposium in Robot and Human Interactive Communication*, pp. 404–409. IEEE.

I. Mlakar, Z. Kačič, and M. Rojc. 2013. Tts-driven synthetic behaviour-generation model for artificial bodies. *International Journal of Advanced Robotic Systems*, 10(10): 344.

A. M. V. Monteiro and T. Pfeiffer. Virtual reality in second language acquisition research: A case on amazon sumerian. *Educational Technologies 2020 (ICEduTech 2020)*, p. 125.

L.-P. Morency, C. Sidner, C. Lee, and T. Darrell. 2005. Contextual recognition of head gestures. In *Proceedings of the 7th international conference on Multimodal interfaces*, pp. 18–24.

S. Mozgai, A. Hartholt, and A. S. Rizzo. 2020. An adaptive agent-based interface for personalized health interventions. In *Proceedings of the 25th International Conference on Intelligent User Interfaces Companion*, pp. 118–119.

R. Niewiadomski, E. Bevacqua, M. Mancini, and C. Pelachaud. 2009. Greta: an interactive expressive eca system. In *Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 1399–1400.

T. Noma, L. Zhao, and N. I. Badler. 2000. Design of a virtual human presenter. *IEEE Computer Graphics and Applications*, 20(4): 79–85.

A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu. 2016. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*.

J. Ostermann. 2002. Face animation in mpeg-4. *MPEG-4 Facial Animation: The Standard, Implementation And Applications*, pp. 17–55.

W.-C. Pang, C.-Y. Wong, and G. Seet. 2018. Exploring the use of robots for museum settings and for learning heritage languages and cultures at the chinese heritage centre. *Presence: Teleoperators and Virtual Environments*, 26(4): 420–435.

S. Pasquariello and C. Pelachaud. 2002. Greta: A simple facial animation engine. In *Soft computing and industry*, pp. 511–525. Springer.

C. Pelachaud. 2002. Visual text-to-speech. *MPEG-4 Facial Animation: The Standard, Implementation And Applications*, pp. 125–140.

P. Petta, C. Pelachaud, and R. Cowie. 2011. *Emotion-oriented systems: the HUMAINE handbook*. Springer.

R. Pieraccini. 2021. *The Handbook on Socially Interactive Agents: 20 years of Research on Embodied Conversational Agents, Intelligent Virtual Agents, and Social Robotics Volume 1: Methods, Behavior, Cognition*, pp. 147–172. ACM Press, 1.

I. Poggi, C. Pelachaud, and F. De Rosis. 2000. Eye communication in a conversational 3d synthetic agent. *AI communications*, 13(3): 169–181.

I. Poggi, C. Pelachaud, F. de Rosis, V. Carofiglio, and B. De Carolis. 2005. Greta. a believable embodied conversational agent. In *Multimodal intelligent information presentation*, pp. 3–25. Springer.

D. Povey, A. Ghoshal, G. Boulianne, L. Burget, O. Glembek, N. Goel, M. Hannemann, P. Motlicek, Y. Qian, P. Schwarz, J. Silovsky, G. Stemmer, and K. Vesely. Dec. 2011. The kaldi speech recognition toolkit. In *IEEE 2011 Workshop on Automatic Speech Recognition and Understanding*. IEEE Signal Processing Society. IEEE Catalog No.: CFP11SRW-USB.

M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. 2009. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, p. 5.

Kobe, Japan.

S. M. Rahman. 2019. An iot-based common platform integrating robots and virtual characters for high performance and cybersecurity. In *2019 SoutheastCon*, pp. 1–6. IEEE.

D. Reidsma and H. van Welbergen. 2013. Asaprealizer in practice–a modular and extensible architecture for a bml realizer. *Entertainment computing*, 4(3): 157–169.

J. Rickel, J. Gratch, R. Hill, S. Marsella, and W. Swartout. 2001. Steve goes to bosnia: Towards a new generation of virtual humans for interactive experiences. In *AAAI spring symposium on artificial intelligence and interactive entertainment*, volume 11.

J. Rickel, S. Marsella, J. Gratch, R. Hill, D. Traum, and W. Swartout. 2002. Toward a new generation of virtual humans for interactive experiences. *IEEE Intelligent Systems*, 17(4): 32–38.

S. Ritter, J. R. Anderson, K. R. Koedinger, and A. Corbett. 2007. Cognitive tutor: Applied research in mathematics education. *Psychonomic bulletin & review*, 14(2): 249–255.

P. S. Rosenbloom, A. Demski, and V. Ustun. 2016. The sigma cognitive architecture and system: Towards functionally elegant grand unification. *Journal of Artificial General Intelligence*, 7(1): 1–103.

B. Rossen and B. Lok. apr 2012. A crowdsourcing method to develop virtual human conversational agents. *International Journal of Human Computer Studies*, 70(4): 301–319. ISSN 10715819. DOI: 10.1016/j.ijhcs.2011.11.004.

S. Scherer, S. Marsella, G. Stratou, Y. Xu, F. Morbini, A. Egan, L.-P. Morency, et al. 2012. Perception markup language: Towards a standardized representation of perceived nonverbal behaviors. In *International Conference on Intelligent Virtual Agents*, pp. 455–463. Springer.

M. Schroder, E. Bevacqua, R. Cowie, F. Eyben, H. Gunes, D. Heylen, M. Ter Maat, G. McKeown, S. Pammi, M. Pantic, et al. 2011. Building autonomous sensitive artificial listeners. *IEEE transactions on affective computing*, 3(2): 165–183.

M. Schröder, M. Charfuelan, S. Pammi, and I. Steiner. 2011. Open source voice creation toolkit for the mary tts platform. In *Twelfth annual conference of the international speech communication association*.

A. Shapiro. 2011. Building a character animation system. In *International conference on motion in games*, pp. 98–109. Springer.

A. Shapiro, A. Feng, R. Wang, H. Li, M. Bolas, G. Medioni, and E. Suma. 2014. Rapid avatar capture and simulation using commodity depth sensors. *Computer Animation and Virtual Worlds*, 25(3-4): 201–211.

H. Shi and A. Maier. 1996. *Speech enabled shopping application using Microsoft SAPI*. PhD thesis.

A. Shoulson, N. Marshak, M. Kapadia, and N. I. Badler. 2013. Adapt: the agent development and prototyping testbed. *IEEE Transactions on Visualization and Computer Graphics*, 20(7): 1035–1047.

M. Si and J. D. McDaniel. 2016. Establish trust and express attitude for a non-humanoid robot. In *CogSci*.

H. A. Simon. 1969. The sciences of the artificial. *Cambridge, MA*.

G. Skantze and S. Al Moubayed. 2012. Iristk: a statechart-based toolkit for multi-party face-to-face interaction. In *Proceedings of the 14th ACM international conference on Multimodal interaction*, pp. 69–76.

B. Snyder, D. Bosnanac, and R. Davies. 2011. *ActiveMQ in action*, volume 47. Manning Greenwich Conn.

R. A. Sottilare, K. W. Brawner, A. M. Sinatra, and J. H. Johnston. 2017. An updated concept for a generalized intelligent framework for tutoring (gift). *GIFTtutoring. org*.

K. Stefanov, B. Huang, Z. Li, and M. Soleymani. 2020. Opensense: A platform for multimodal data acquisition and behavior perception. In *Proceedings of the 2020 International Conference on Multimodal Interaction*, pp. 660–664.

M. Stone and C. Doran. 1997. Sentence planning as description using tree adjoining grammar. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, pp. 198–205. Association for Computational Linguistics.

G. Stratou and L.-P. Morency. 2017. Multisense—context-aware nonverbal behavior analysis framework: A psychological distress use case. *IEEE Transactions on Affective Computing*, 8(2): 190–203.

S. Sutton, R. Cole, J. De Villiers, J. Schalkwyk, P. Vermeulen, M. Macon, Y. Yan, E. Kaiser, B. Rundle, K. Shobaki, P. Hosom, A. Kain, Johan, J. Wouters, D. Massaro, and M. Cohen. 1998. Universal Speech Tools: The Cslu Toolkit. *in proceedings of the international conference on spoken language processing (ICSLP)*, 7(September 2014): 3221 – 3224.

W. Swartout, D. Traum, R. Artstein, D. Noren, P. Debevec, K. Bronnenkant, J. Williams, A. Leuski, S. Narayanan, D. Piepol, et al. 2010. Ada and grace: Toward realistic and engaging virtual museum guides. In *International Conference on Intelligent Virtual Agents*, pp. 286–300. Springer.

W. R. Swartout, J. Gratch, R. W. Hill Jr, E. Hovy, S. Marsella, J. Rickel, and D. Traum. 2006. Toward virtual humans. *AI Magazine*, 27(2): 96–96.

W. R. Swartout, B. D. Nye, A. Hartholt, A. Reilly, A. C. Graesser, K. VanLehn, J. Wetzel, M. Liewer, F. Morbini, B. Morgan, et al. 2016. Designing a personal assistant for life-long learning (pal3). In *The Twenty-Ninth International Flairs Conference*.

T. B. Talbot and A. S. Rizzo. 2019. Virtual standardized patients for interactive conversational training: A grand experiment and new approach. In *Exploring the Cognitive, Social, Cultural, and Psychological Aspects of Gaming and Simulations*, pp. 62–86. IGI Global.

P. Taylor and A. Isard. 1997. Ssml: A speech synthesis markup language. *Speech communication*, 21(1-2): 123–133.

P. Taylor, A. W. Black, and R. Caley. 1998. The architecture of the festival speech synthesis system. In *The Third ESCA/COCOSDA Workshop (ETRW) on Speech Synthesis*.

M. Ter Maat and D. Heylen. 2011. Flipper: An information state component for spoken dialogue systems. In *International Workshop on Intelligent Virtual Agents*, pp. 470–472. Springer.

K. R. Thórisson, T. List, C. Pennock, and J. Dipirro. 2005. Whiteboards: Scheduling Blackboards for Semantic Routing of Messages & Streams. Technical report. http://www.naradabrokering.org/.

D. Traum, A. Jones, K. Hays, H. Maio, O. Alexander, R. Artstein, P. Debevec, A. Gainer, K. Georgila, K. Haase, et al. 2015. New dimensions in testimony: Digitally preserving a holocaust survivor's interactive storytelling. In *International Conference on Interactive Digital Storytelling*, pp. 269–281. Springer.

S. Ultes, L. M. Rojas Barahona, P.-H. Su, D. Vandyke, D. Kim, I. n. Casanueva, P. Budzianowski, N. Mrkšić, T.-H. Wen, M. Gasic, and S. Young. July 2017. PyDial: A Multi-domain Statistical Dia-

logue System Toolkit. In *Proceedings of ACL 2017, System Demonstrations*, pp. 73–78. Association for Computational Linguistics, Vancouver, Canada. http://aclweb.org/anthology/P17-4013.

M. Valstar, T. Baur, A. Cafaro, A. Ghitulescu, B. Potard, J. Wagner, E. André, L. Durieu, M. Aylett, S. Dermouche, et al. 2016. Ask alice: an artificial retrieval of information agent. In *Proceedings of the 18th ACM International Conference on Multimodal Interaction*, pp. 419–420.

H. Van Welbergen, R. Yaghoubzadeh, and S. Kopp. 2014. Asaprealizer 2.0: The next steps in fluent behavior realization for ecas. In *International Conference on Intelligent Virtual Agents*, pp. 449–462. Springer.

H. Vilhjálmsson, N. Cantelmo, J. Cassell, N. E. Chafai, M. Kipp, S. Kopp, M. Mancini, S. Marsella, A. N. Marshall, C. Pelachaud, Z. Ruttkay, K. R. Thórisson, H. Van Welbergen, and R. J. Van Der Werf. 2007. The Behavior Markup Language: Recent Developments and Challenges. Technical report. http://wiki.mindmakers.org/projects:BML:main.

J. Wagner, F. Lingenfelser, T. Baur, I. Damian, F. Kistler, and E. André. 2013. The social signal interpretation (ssi) framework: multimodal signal processing and recognition in real-time. In *Proceedings of the 21st ACM international conference on Multimedia*, pp. 831–834.

R. Wallace. 2003. The elements of aiml style. *Alice AI Foundation*, 139.

Web3D, 2006. All H-Anim Standards — Web3D Consortium. https://www.web3d.org/standards/h-anim.

Ö. N. Yalçın and S. DiPaola. 2019. M-path: a conversational system for the empathic virtual agent. In *Biologically Inspired Cognitive Architectures Meeting*, pp. 597–607. Springer.