Chapter 4
Simulation Technologies

Ben Hawkes

4.1 Introduction

An exploration of the technologies employed in creating and deploying assessment simulations can only cover a subset of this expansive topic. Despite this, it is hoped that this chapter will enable the reader to understand some of the technologies involved in developing and deploying assessment simulations, the advantages and disadvantages of such technologies, and ultimately to make informed decisions about simulation development, regardless of the reader's role in that process.

The focus of this chapter is on assessment simulations that fulfill three criteria:

- Can be deployed to candidates in both *proctored and unproctored* settings. Deploying simulation assessments to candidates via the Internet imposes certain constraints on designers which are explored further on.
- *Capture candidates' responses*. Many simulations react to the user's input without capturing and storing that input. For assessment for selection, however, we need to be able to capture a candidate's response in order to score it or compare it to the responses of others.
- *Report those responses back* to a database such as a learning management system (LMS) or applicant tracking system (ATS). When used for online assessment, simulations are typically delivered from a platform that initiates the simulation, and then captures and stores the candidate's responses and scores. This may be a stand-alone assessment platform, or an ATS or an LMS.

Care has been taken to avoid technical jargon, or, where it is necessary for the understanding of assessment technologies, to explain it. For the purposes of this chapter, the starting point is the "script": the text of the assessment, perhaps created by an industrial–organizational (I–O) psychologist or other assessment professional, already trialed. The process of creating simulations has been segmented into three stages: content creation (creating the components of the simulation), authoring (tying

B. Hawkes (✉)
Kenexa, an IBM Company, London, UK
e-mail: ben.hawkes@uk.ibm.com

M. Fetzer, K. Tuzinski (eds.), *Simulations for Personnel Selection,*
DOI 10.1007/978-1-4614-7681-8_4, © Springer Science+Business Media New York 2013

Table 4.1 Types of simulation and their applications

Type of simulation	Example applications
Character-based simulation	Multimedia situational judgment test (SJT), presenting candidates with short scenes of customer interactions and asking them to select the best and worst response from a list of four or five options
Desktop simulation	Call-center simulation, presenting users with a facsimile of a call-center operator's screen. Electronic inbox or "e-tray" exercise, presenting candidates with documents and data to be analyzed and acted upon
Virtual environment (VE)-based simulation	Users interact with and manipulate representations of three-dimensional (3D) objects, from individual objects to complex machines all the way to full virtual environments

all of those components together), and deployment (putting the simulation online and connecting it to other online services such as an ATS). These three stages are not mutually exclusive: although some tools have a role in only one stage of the process, many of them span two or even all three stages.

Throughout the chapter, reference is made to various technologies, software, and websites. These are not intended as endorsements: the technologies available to simulation creators and users are varied, and those that are referenced in this chapter are provided as exemplars. Many of the more commonly seen technologies are listed in Sect. 4.5.

This chapter will focus on three common types of assessment simulation, outlined in Table 4.1.

4.2 Content Creation

In this context, "content" refers to everything that the candidate will see and hear during the course of the simulation, sometimes referred to as the collateral or assets. These assets could be text, images, animation, video, or audio.

To create images, simulation developers have several options. The creation of image content may be as simple as commissioning a series of photographs or buying suitable images from a stock library, or images may be created by a graphic designer in order to create the user interface. The remainder of this section focuses on the creation of other forms of multimedia: animation, video, and audio.

Fig. 4.1 Example of two-dimensional animation

4.2.1 Character-Based Simulations

To design character-based simulations, designers are faced early on with the choice of using "live action" (i.e., video of real people) versus animation. So what factors should designers consider in making this choice between these two media?

4.2.1.1 Animation

Originally, animation was a painstaking process: each second of footage would require 25 individually drawn images, to be shown one after another. Nowadays, through the assistance of animation software, such efforts are not necessary. At the very least the software streamlines the workflow, but in many instances it can create much of the animation automatically.

At the high-end of animation, of the quality shown in Disney Pixar movies, animators still use a painstaking approach. That approach comes at a high cost: the estimated budget for the film *Up* was $ 175 million (IMDB 2012). At 96 min long, that suggests a rate of $ 1.8 million per minute, or around $ 30,000 per second of animation!

However, most simulation designers are unlikely to have the budget of Disney Pixar. Instead, footage is commissioned from animators using software such as Autodesk's Maya and 3ds Max which have become almost "industry standard" for professional three-dimensional character animation, and Toon Boom, RETAS, and Toonz which have held a similar position amongst two-dimensional animation packages. Figures 4.1 and 4.2 show examples of two-dimensional and three-dimensional animation, respectively.

These programs provide a "blank canvas" with virtually no limitation to what can be created. To create a high-quality output from such software, however, requires extensive experience and knowledge, and this is reflected in the time and costs of creating animation. It is extremely difficult to give even a "ballpark" estimate on animation costs as it depends on the complexity of the scene, character movements,

Fig. 4.2 Example of three-dimensional animation

Fig. 4.3 Example simulation content produced using CodeBaby, an off-the-shelf animation tool

lip-synching, and so on. Hawkes (2012) supplied an identical brief to a selection of animation agencies in the USA, UK, and India. Their quoted costs ranged greatly, with the most expensive quote at ten times the cost of the least expensive.

Over the past 5 years, off-the-shelf character animation software designed for non-animators has become capable of producing high-quality outputs through a simple process of "drag and drop" editing. In the same way that presentations can be assembled by dragging and dropping shapes, text, and pictures, so animation can be created by dragging and dropping characters, backgrounds, and even movements. Figure 4.3 shows an example of the output of one tool, CodeBaby. In this example, off-the-shelf characters have been combined with a custom-built background environment.

However, this ease of use comes at a price. With these simpler-to-use tools, such as CodeBaby, Moviestorm, Xtranormal, and Muvizu, the designer's choices are restricted: characters, of a particular style, can only be customized to the extent permitted—or programmed—by the vendor.

However, a significant advantage of off-the-shelf animation tools is that the production time can be much shorter. One minute of animation might take just a few

hours to create, whereas using professional animation software, one minute of animation might take days or even longer to create. There are various reasons for this. First, off-the-shelf tools often come stocked with a "library" of character movements and facial expressions that speed up the animation process. One of the more time-consuming stages in character animation is synchronizing mouth and lip movements to speech. Many of these off-the-shelf tools handle that automatically. Finally, such software packages often come with ready-to-use characters, environments, and objects, or at least offer the opportunity to download these for a cost from the vendor and third parties.

Between the consumer and professional packages lie other software such as Reallusion's iClone and DAZ Studio. Although not without their limitations, these offer much of the customization offered by professional software, but with greater ease-of-use.

Regardless of the style of animation that is selected, the characters must have a voice. Again, developers are faced with a choice: to use "real" people or to use computer generated speech, generally known as Text To Speech (TTS).

The quality of TTS has increased significantly over the past 10 years. Although it used to clearly sound machine-generated, now TTS vendors are able to create very clear human-like speech. However, developers of simulations are faced with a particular challenge when trying to embed emotional content into speech. While some TTS technologies allow text to be "marked up" to indicate where words and syllables should be stressed or delivered in a particular style, this still does not rival the expressive range of a human actor.

However, TTS still has a place for simulation developers. When prototyping a new simulation, it may not be cost-effective or practical to bring in voice actors, particularly if the spoken content of the simulation is likely to be changed frequently before its final incarnation. In this case, TTS can be used to create the spoken content rapidly and at low cost.

4.2.1.2 Video-Based Simulations

Video-based, live-action content offers several advantages over animated content. First, by capturing footage of real people, the simulation presents real behavior, including the subtle movements and expressions that we as observers rely on to understand the thoughts and feelings of others.

A second advantage comes with the potential speed of producing live-action footage. For certain types of material, the video production process can be much quicker than producing an equivalent animation. For example, a live-action version of a single two minute interaction taking place between two employees could be shot and edited in the space of several hours. To re-create this using animation, depending on the style of animation chosen, could take several days to produce. Animating the characters might only take a fraction of this time, with the remainder taken up by designing the characters and environment, and recording the voice audio for the animation to use.

However, live-action video does present some disadvantages. Firstly, although the time required to shoot simple interactions—such as the employee interaction mentioned—may be short, more complex interactions and situations can take significantly longer, when one factors in script learning, rehearsal, multiple camera angles, reshooting, and editing.

Secondly, animated sequences can more quickly and easily be updated than live-action footage, allowing rapid revision of existing test items to reflect changing psychometric (e.g., updated items), aesthetic (e.g., new branding), and localization needs (e.g., redubbing the animation into different languages). Making similar updates to live-action footage could well require extensive—and possibly expensive—reshoots.

A third disadvantage is that shooting video on location may present practical challenges. In the case of video being shot for a customer service simulation, if the video is being shot within a retail store then consideration must be given to planning the shoot to minimize the potential disruption that having a film crew may have on the day-to-day operations of that store. In other instances, it may not be practical to shoot a live-action version of the scenario. These instances might include the portrayal of unsafe working practices or catastrophic events: animation may be the only realistic option.

The fourth—and perhaps for many simulation developers, the most important—disadvantage of shooting live-action video is that the costs can become significantly higher than animation—perhaps to the extent that live action is not a feasible possibility. For example, a military simulation might require footage of many moving soldiers, ground vehicles, and aircraft. A live-action version of this might take months to plan and cost tens or even hundreds of thousands of dollars to shoot. The alternative of creating an animated version of this footage could be completed in shorter timescales and at a greatly reduced budget.

Of course, many simulations are less ambitious in their scope, and will require nothing like the budget and planning of a large-scale military simulation. Fox (2010) provides a list of 25 factors that impact the cost of corporate video production, including people (e.g., director, crew, actors, and editors), location, and equipment. Simulation developers can use a similar list to compare potential costs of animation versus live-action production.

4.2.2 Screen-Based and Desktop Simulations

Often, simulations need to re-create the working environment of a role that is either entirely PC-based, or are likely to use a PC much of the time—for example, call-center operators. In that case, the visual design of the simulation might simply be a re-creation of a screen interface.

In desktop simulations, the designer may choose to create a simple interface with a familiar graphical user interface (GUI) as one might see on a Windows or Mac "desktop." Figure 4.4 shows an example of this.

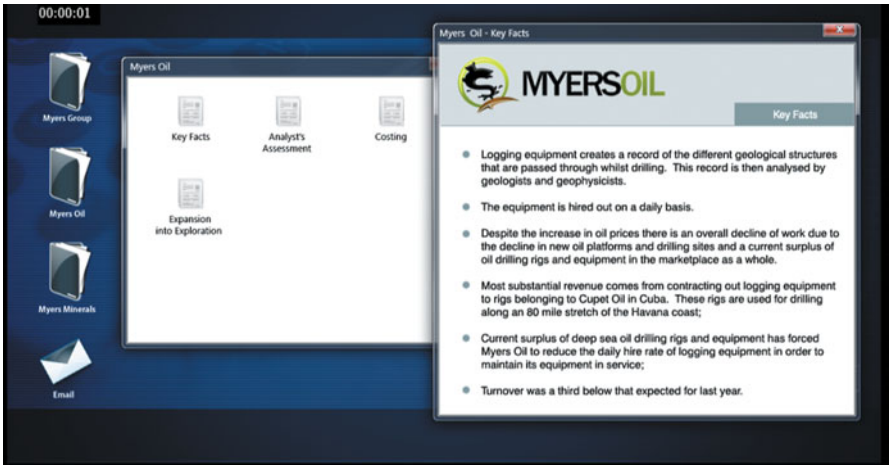

Fig. 4.4 Example of desktop simulation

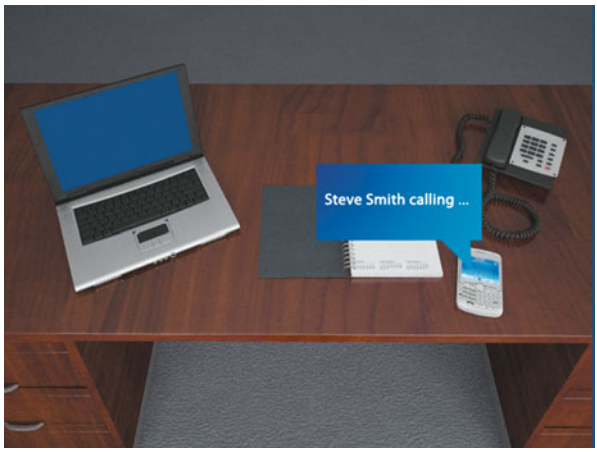

Fig. 4.5 Example of a detailed desktop user interface

Alternatively, the desktop simulation may be more complex. Figure 4.5 shows an in-tray simulation taking place in a detailed three-dimensional office environment.

The choice of whether to use a fully rendered three-dimensional office, or simply to present a computer desktop may be as much to do with aesthetic choice as it is to do with budget. Regardless of the choice, care must be taken that the interface is easy to use. If users find the interface cumbersome or unfamiliar, then there may be a risk of introducing error into the assessment (Maxion and Reeder 2005).

4.2.3 Virtual Environment-Based Simulations

Virtual environment (VE)-based, or "virtual world" simulations present candidates with a re-creation of real-world objects, environments, and characters. They can

Fig. 4.6 Example of a virtual environment-based simulation assessment

often interact with these re-creations in realistic ways—for instance, disassembling a machine part or walking around a three-dimensional facsimile of a factory. Characters in the VE can be either entirely automated and computer-controlled, or they can be controlled by other people acting as virtual puppeteers (Fig. 4.6).

Although character-based simulation and desktop simulations may require modest animations and graphics, VE-based simulations often have far more complex requirements, for a three-dimensional virtual world needs to be populated with three-dimensional characters, environments, and props. It is the creation of these assets that often accounts for a large proportion of the development budget and time for VE-based simulations.

In many instances, however, these assets do not need to be created from scratch. Libraries of third-party characters, equipment, and other assets exist which can be used "off-the-shelf", or customized to meet the needs of the simulation. Typically, these assets are sold under a license, so care must be taken that the license permits the use for the asset that the simulation developer has in mind.

4.3 Authoring and Deployment

Authoring a simulation is the process by which the content (video, animation, audio, text, and images) is assembled into the finished product. It is similar in its objectives to the process of assembling video, animation, and other content into a presentation. The difference being that a simulation, as well as presenting information, also captures and stores candidates' responses.

Deployment of the simulation is the final stage of its development—the equivalent of letting it "into the wild." However, often this is the stage that simulation designers will consider first. This is done for a simple reason: the technology and methods for deploying the simulation will determine many of the choices that can be made during the content creation and authoring processes.

This section begins with an overview of some of the more commonly used tools to author and deploy assessment simulations before looking in detail at some of the challenges that are faced when we aim to deploy simulations in the context of online unproctored assessment.

4.3.1 Software for Authoring and Deployment

To maximize the reliability of any assessment—paper-based or computer-based—it is necessary that each candidate receives a standardized experience. That is not to say it should be identical: the principles of computer adaptive testing (CAT) are well established enough that we can be confident that candidates responding to different questions can still be considered to have taken the same test. Also, despite the multiple possible combinations of browser (Microsoft Internet Explorer, Google Chrome, Apple Safari, Mozilla Firefox, Opera, etc.) and operating system (Windows, Mac, Linux, Android, iOS), every year many millions of online assessments are delivered in a standardized and reliable form.

However, "traditional" assessments comprise text and graphics. Assessment simulations on the other hand may combine text, graphics, animation, video, and interactivity. So how do simulation creators ensure that the candidates' experiences are standardized?

Web pages are written using a language called Hypertext Markup Language (HTML). This instructs the browser software how to lay out the screen, what size and font to use for text, where to position images, and so on. However, the most common version of HTML in use—Version 4—does not itself provide the functionality that many simulations require. For example, it has limited capability to deliver video and audio, and it does not natively support animation.

To get around these limitations, multimedia and other rich Internet applications (RIA) use browser "plug-ins." These are small downloads that extend the capability of the browser and allow them to run software that has been created in other languages, not just HTML.

The plug-in that most Internet users will have encountered is Adobe Flash. Flash works within Microsoft Internet Explorer, Firefox, Safari, and other Internet browsers to deliver multimedia and interactive content. Over 99 % of PCs worldwide have Flash installed, although on mobile devices (phones, tablets) the percentage is substantially lower, around 50 % (Adobe Systems Incorporated 2012).

To understand the role of Flash and similar technologies, it might be useful to think back to a time when the World Wide Web was not supported by these technologies. As recently as 10 years ago, multiple conflicting standards for deploying animation, video, and even sound meant that the experience of using the Internet was far less seamless and transparent than today. Websites used a variety of standards and proprietary formats to deliver multimedia content. If the user's browser was unable to display the content, then they might be invited to download the appropriate plug-in.

As bandwidth increased and PCs became more powerful, a standard method of delivering rich, interactive content was needed, and Adobe (then Macromedia) Flash was launched (Macromedia 2002) and has since become the almost ubiquitous browser plug-in we see today.

Other plug-ins are commonly used to run simulations. Java, published by Oracle Corporation, and Silverlight, published by Microsoft, offer many capabilities to simulation developers that are similar to Flash. Lively discussions of the merits of Flash, Java, and related technologies can be found on the Internet (Brandt 2012; Draney 2012; Franco 2010; Le Grecs 2012), and simulation designers looking for guidance are unlikely to find a definitive answer. However, if Flash is ubiquitous and supported on so many platforms, why should simulation developers even consider other technologies for deployment?

One reason is that despite Flash being considered a "cross platform" language usable by different types of computers and operating systems not all mobile devices (phones and tablets) support Flash. In fact, no devices running Apple iOS—including iPad, iPhone, and iPod—run Flash within their browsers, nor are ever likely to (Jobs 2010). Support on other mobile devices—those running Android and Windows operating systems for example—is commonplace, but with Adobe's 2011 announcement that Flash will no longer be developed for new mobile devices (Adobe Systems Incorporated 2011; Winokur 2011), it is likely that additional technologies will take the place of Flash as the most common method of delivering media-rich content, at least on mobile devices.

One of these technologies is the latest version of HTML, HTML5. Although the standard is not due to be finalized until 2014, most up-to-date PC, Mac and mobile-based browsers support it. The difference with this latest version is that animation, audio, and video are supported natively in the browser. In other words, the browser does not have to download a plug-in like Flash to deliver media-rich content. Fewer plug-ins mean fewer potential compatibility issues and, hopefully, fewer problems for the end-user.

However, compared to Flash, HTML5 has a limited set of features and capabilities. Graphics, audio, and video are well supported, but client-side applications—which are often needed to run simulations—receive inconsistent support from browsers (Deep Blue Sky 2012). Having said that, many simulations can be developed and run quite adequately in HTML5. However, more sophisticated simulations of the type often programmed in Flash may not be supported by HTML5.

There are ways of extending the functionality of HTML. One way is through the use of JavaScript. This is a programming language that runs within the browser and is capable of creating interactions from simple click and drag up to fully functioning, sophisticated word processors.

The choice of platform—Flash, Java, HTML, and so on—must be made very carefully. Designers should take into account the desired functionality and user experience for the simulation. In the interests of making the simulation as accessible as possible, designers should also consider whether users will have the necessary plug-in installed and whether that plug-in can even be installed at all. At the time of writing, in early 2013, simulation designers wishing to ensure that their assessments

will run on iOS-based devices as well as older PC-based browsers may have to consider whether two versions of each simulation—one in Flash and one in HTML5—must be created to maximize the accessibility to the pool of candidates, and reduce the likelihood of candidates being unable to take the assessment. Over time, market penetration of devices and plug-ins change, so simulation designers should consult up-to-date statistics before selecting which technologies to use to create their assessments. Some sources for these statistics are listed at the end of this chapter.

4.3.1.1 Specialized Tools

Aside from the generic authoring tools such as Flash, there are many tools that are specifically designed for creating simulations for learning and development. Weiss (2012) counts over 140 of them. Many of those same tools can also create simulations suitable for assessment for selection.

These tools vary enormously in their capabilities, objectives, and look and feel. We have chosen a few examples of each style of simulation technology, but as iterated in the introduction to this chapter, their inclusion is in no way an endorsement of the product.

Many specialized simulation development tools offer the ability to export the finished simulation to Flash, Java, Silverlight, or many other formats discussed above. This enables the simulation to be deployed in the same way as an application that uses those formats. The advantage of using a specialized simulation authoring tool is that these technologies offer greater access to the non-programmer. Instead of writing lines of code, users can create simulations by dragging and dropping elements onto flowcharts. This presents two possible advantages: simulations can be built more quickly, and the development process is often more accessible to non-technical developers.

4.3.1.2 Linear Versus Branching Simulations

At their simplest, a simulation may be entirely linear: a video plays, question one appears, a second video plays, question two appears, and so on. Multimedia SJTs often use this format. This is a format that even simple simulation authoring software can produce.

Other simulations are nonlinear in that the candidate's responses are used by the simulation to determine which part they see next, almost like a "Choose Your Own Adventure" book. For example, in a sales simulation, the candidate could be given the choice of (a) asking more questions of the simulated customer, or (b) to present recommendations to the customer. The customer would then react realistically according to the candidate's choice, and the candidate is presented with further choices to make.

A simulation that offers branching after *every* stimulus offered to the candidate would rapidly become unwieldy. For example, if the candidate watches one video and

is given three choices, each one leading to another video which in turn leads to three more choices, then for the fifth stage some 243 separate videos would be required. Instead, most branching simulations avoid this exponential growth by limiting the number of decision points or by limiting the number of options at each decision point.

Some authoring tools—for example Articulate Storyline and ClicFlic—provide a graphical interface with which to design such branching simulations. This interface often takes the form of a flowchart and the branching can be designed visually rather than using programming code. Using a graphical interface can make the simulation authoring process more accessible to nonprogrammers and reduce development time.

A branching simulation presents certain psychometric challenges: with its potential of thousands of different permutations of question order and candidate experience, the trialing and validation process needs to be carefully considered. This is not an issue that will be discussed here, suffice to say that any assessment, branching simulations included, should be trialed and validated to ensure that its use is appropriate and defensible.

4.3.2 Challenges Faced in Deploying Assessment Simulations

The use of simulations to deliver remote, unproctored assessments presents a series of challenges compared to the use of simulations in a proctored and controlled setting. In a controlled setting, administrators have control over the hardware, software, and testing environment. They also can offer direct and prompt IT support to candidates. Compare this to unproctored simulations which are being accessed by untrained users of unknown ability through a variety of browser software running on various operating systems and hardware, via Internet connections of varying quality, speed, and bandwidth.

Let us consider each of those constraints in turn, as each of them has an impact upon the design of assessment simulations.

4.3.2.1 Accessibility and Disability

We must be careful to consider the needs of candidates with disabilities. A visually rich simulation may well offer the sighted candidate an immersive, engaging experience, but unless suitable accommodation is made, there is a risk that candidates with visual impairments are disadvantaged. Equally, another simulation might make use of videos of conversations between characters, but without suitable accommodation, candidates with hearing impairments could be disadvantaged. So while simulations may well offer increased accessibility, creators must be careful to ensure that all candidates are assessed equally, regardless of computer literacy or disability.

In designing any assessment, simulations included, we must consider its accessibility. Commonly considered to refer to candidates with disabilities, accessibility

also is relevant to other candidate groups, including older people and candidates with language impairment, or simply candidates who rarely, if ever, use a computer. Legislation often provides assessment publishers and users with a set of requirements that must be met. However, these requirements—and their very existence—are not universal, varying between countries and regions, and sometimes only providing the briefest of guidance. The Web Accessibility Initiative (WAI) of the World Wide Web Consortium (W3C) provides extensive guidelines and support materials to help designers understand and implement web accessibility (W3C 2012), and these are strongly recommended to designers. Simulation developers and users may also want to refer to the "Section 508" guidelines (United States Government 2011). Section 508 requires Federal agencies to take into the account the needs of all end-users when implementing electronic or other forms of information technology. Although targeted toward US Federal agencies, the guidelines may prove useful for any developer or user of simulations and assessments.

4.3.2.2 Processor Speed and Memory

The speed of the processor governs how fast the hardware can interpret and process instructions. A processor that is too slow for the software might run that software slowly, or even refuse to run it at all. The memory limits how much data can be stored. This is an issue that designers of internet-based simulations might come up against if, for example, the simulation required the computer to display full-screen, high-quality video. Bear in mind though that the candidate may have other software running on their computer when they access the simulation. Even though the simulation may only have modest requirements, it may be competing for resources with other software that might be running.

4.3.2.3 Screen Resolution

Screen resolution refers to the number of individual dots, or pixels, the screen contains. Typically, this is expressed by the number of pixels in each direction, e.g., 640×480 or 1024×768. Higher resolution means that higher detail can be displayed. As the numbers of pixels in both direction doubles, the amount of information contained on the screen quadruples: a screen of 1280×960 contains four times the data as a screen of 640×480.

Desktops and laptops typically have higher resolution than tablet devices, which have higher resolution than mobile phones. This is not always true, however: recent mobile phones are rivaling laptop displays for resolution.

4.3.2.4 Screen Size

Along with the resolution, simulation designers must take into account the physical screen size of the candidate's device. This is independent of screen resolution: a desktop display may be 22 in. diagonally, a tablet screen 12 in., and a mobile phone

screen 3 in. Simulation designers must consider the sizes of the candidates' screens and design accordingly. That is not to say that all simulations must be designed to run on the smallest screen available, however: web pages can be programmed to detect the browser, operating system, and device upon which they are being displayed and, if the requirements are not met, to block its use and recommend to the candidate that he or she should use an alternative device.

4.3.2.5 Interface

The majority of desktop- and laptop-based software has to date been designed for keyboard and mouse (or touchpad). Users of these interfaces often become accustomed to the standard techniques of drag and drop, double-clicking, right-clicking, and so on. However, even with desktop computers there are differences. Mice used on Apple computers tend to have a single button. An instruction to a Mac-using candidate to "right click" in a simulation could well be met with confusion.

Many modern phones and tablet devices use touch screens. Early devices would allow one touch at a time. More recently, devices allow five or more simultaneous touches, giving the user the ability to literally physically manipulate on-screen information. This new type of user interface offers simulation designers new opportunities to engage and interact with users. On the other hand, many tablets and smartphones use "smart keyboards." These are on-screen keyboards that appear, often automatically, when the user is required to type information. As they are on-screen, simulation designers must be careful to ensure that the keyboard does not obscure important information. If they do, and if the user is forced to alternately minimize and restore the keyboard, they could well be at a disadvantage compared to real keyboard users who get to see the screen the entire time.

4.3.2.6 Sound

While most desktop and laptop computers, tablets and mobile phones have sound capability, this does not necessarily mean that they are available or accessible to the simulation. PC speakers may be unplugged, sounds may be muted, or sounds may be inaudible against loud background noise. If a simulation is dependent on audio content, candidates should be made aware of this at the beginning of the assessment, and given the opportunity to listen to sample audio to ensure that their device is playing the sound, and that it is audible in the environment. If users cannot hear the sound, they should be given the opportunity to use the simulation from another location or device.

4.3.2.7 Internet Connectivity

While broadband is increasingly commonplace, it is by no means universal. At the end of September 2012, 75.4 % of US households had fixed broadband (Mastrangelo 2012), with approximately 66 % of American adults having broadband connections

at home (Zickuhr and Smith 2012). The number of adults who only have dial-up access at home is less than 3 %, and declining year after year (Zickuhr and Smith 2012).

So is it safe for simulation developers to design solely for broadband? Doing this could potentially limit the target audience's accessibility to the simulation. However, where a simulation is using audio, video, and animation, it may simply not be possible to create a version that will work via a dial-up modem.

Instead, simulation designers should design simulations for low-speed bandwidth use and, once created, they should be tested at those speeds. Many users of corporate and educational IT networks experience data speeds of 100 Mbps, 25–50 times faster than typical domestic broadband speeds. Testing the download speed of a simulation in a corporate environment will not emulate the experience of the typical home user.

4.3.2.8 Security

Whenever we use the Internet, some processes are carried out by the user's own computer (client-side), and some process are carried out by the host computer that the user has connected to via the Internet (server-side).

If we access a Flash-based simulation, then our computer downloads the Flash application and runs it. That application not only provides the user experience, but it also transfers the candidate's responses back to the server. But now that the simulation application has been downloaded client-side, it may be possible to disassemble that simulation, unpacking all of its contents and programming. This information could then be used by a candidate to cheat, or shared amongst other potential candidates. The simulation could even be "reverse engineered," or modified, to send back a series of correct responses to the server, enabling the candidate to receive a perfect score. This scenario does not just apply to Flash-based simulations, but any simulation that runs client-side code, including Java, Silverlight, and other technologies discussed.

Granted, there will only be a small proportion of candidates with the technical capability to do this. However, where a simulation might be administered to tens or even hundreds of thousands of candidates, there remains a possibility that its security could be breached.

There are steps that simulation developers can take to improve the security of simulations. The programming code can be "obfuscated," making it more difficult to disassemble and read. This may well deter some, but better security comes from minimizing the amount of data and information stored on the candidates' computers. This means putting as little data "client-side" as possible.

For example, a simple simulation might present a series of options to the candidate and they answer with a multiple-choice response. If that response is scored as correct or incorrect within the client-side Flash application itself—i.e., on the candidate's own computer—then the candidate essentially has the scoring key stored on their computer, albeit in a coded or obfuscated form. Potentially, that scoring key could be extracted, giving the candidate a list of correct answers. But instead, if the Flash application simply sends the raw responses back to the server for the server-side software to score, then the scoring key remains secure.

Through simple techniques such as this—as well as a number of far more sophisticated techniques—we can minimize the risk that simulated assessments become compromised.

4.3.2.9 Deploying Internationally

If the simulation is to be deployed to countries outside of the USA, then designers should investigate broadband speeds within those countries as they may not match those commonly seen in the USA. In some countries, an alternative, parallel assessment may need to be offered to accommodate candidates without access to broadband.

For those candidates without access to broadband, video or animation may not be a possibility: a short video of 30 s duration could take many minutes to download. An alternative, without resorting to a text-only assessment, might be to combine audio and still pictures, or even text and still pictures. The benefit of the latter approach is that applicants can then read and scroll through the text at their own pace, returning to any parts of the content to reread.

Simulation developers may run into technical challenges when deploying simulations into other countries. If the simulation is hosted—i.e., physically stored—in one country, say the USA, then users who are some distance away (e.g., in China) may experience a greater lag or delay than USA-based users would experience. Possibly, they may also experience a greater likelihood of their connection being lost. To overcome this issue, some simulation publishers host the simulation "in the cloud." This means that the simulation is hosted in multiple physical locations around the world. In a process that is completely transparent to the candidate, when they access the simulation they download it from the nearest host, instead of from a host located on the other side of the world.

4.3.3 Assessing in Virtual Environments

Some simulations can provide a richly detailed and immersive facsimile of the world in which users can engage, interact, and demonstrate their competencies, skills, and knowledge. Virtual environments are example of these types of simulations. The simulation designer has essentially a blank canvas to create an immersive environment in which users can see, hear, and experience situations which would not be practical to model in real life. Virtual environments are used extensively in military and medical training and assessment for this reason.

So if this is the case, why do we not see more VEs used for unproctored assessment? One of the reasons appears to be the learning curve. While VEs can offer a rich immersive environment, this can come at a cost to their usability: when people want to move around or manipulate objects in the real world, they simply do it—they walk and they pick things up. In a VE, however, these actions must be made through

the interface between the user and the virtual world—typically the computer screen, mouse, and keyboard. Despite efforts of VE creators, this added layer decreases the usability of those environments compared to the real world.

One of the most extensively researched VEs is Second Life, a virtual world platform launched in 2002 by Linden Labs (Linden Research, Inc. 2012). Like other virtual worlds, it offers users an expansive world in which users take on the form of a virtual character or "avatar," allowing them to travel throughout the VE, interact with other users, and create or manipulate objects.

Some studies have compared real-world and virtual-world behavior in VEs such as Second Life, and found significant differences (Richardson et al. 2011; Satalich 1995). These findings suggest that the assessment of a candidate's performance in a VE is not necessarily an accurate measure of the performance that they might demonstrate in the real world. One of the reasons for this may be the steep learning curve of VE platforms (Mennecke et al. 2008). Assessing in a VE may introduce test error by favoring those candidates who have more experience of video gaming. Richardson et al. (2011) found that gaming experience was related to performance in desktop VEs. In addition, Ausburn (2012) found that performance was affected by gaming experience, gender, and age.

A source of error in VE simulations is the difficulty that some candidates have with learning the user interface and subsequently using that interface to navigate through the virtual three-dimensional world. This challenge appears to account for some of the differences in performance found by Richardson et al. (2011). One technique to eliminate or minimize the challenge of navigation is to remove that control from the candidate entirely. Instead of the candidate walking the character around the environment, he or she is taken around it as if they are "on rails." This has the result that the user still experiences some of the immersive quality of the VE, but with increased ease-of-use.

A second benefit of the "on-rails" approach is that the candidate experience can be controlled and standardized. In a VE in which a candidate is free to move around as they please, there is a risk that they may miss vital information. For example, if the assessment simulation takes place in a virtualized office environment and the candidate is free to move from one location to another, then they might rapidly navigate through the virtual office without noticing characters and information that could later prove to be instrumental in that candidate's approach to the task. Using an "on-rails" approach allows simulation designers to ensure that all candidates receive the same experience.

Figure 4.7 shows an example of an "on-rails" simulation. In this simulation, characters can interact with the patient and medical equipment through simple mouse clicks, increasing the ease-of-use for candidates.

Once the issue of usability is resolved, assessing using VE simulations offers the opportunity to move away from simple question/answer models of assessment and introduce many other ways of recording candidates' behavior. For example, a "traditional" assessment of spatial thinking may present candidates with a still image of a three-dimensional object, and five different images of unfolded versions of that object. They are then required to choose which one of the unfolded versions is the

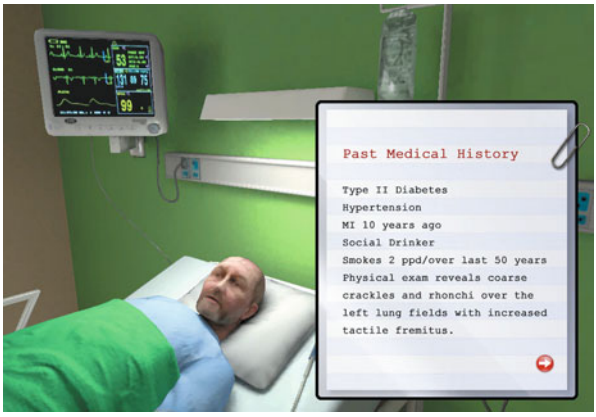

Fig. 4.7 An "on-rails" virtual environment-based simulation

same as the three-dimensional version. Once they have chosen, we have two data points that we can use for assessment: which answer option the candidate chose, and how long it took to respond.

However, if this same item is delivered in a VE-based simulation, we have the potential of measuring many more data points. If candidates are able to rotate the three-dimensional object, we can measure how many times the object was moved, what directions it was oriented in, or whether it was manipulated at all. We could also track mouse movement: did the candidate's mouse pointer hover over various answer options before clicking one, or was it moved directly to the chosen option? We could use eye-tracking technology to measure where on the screen candidates were looking: how long did they spend looking at the three-dimensional object? Did they look at each option equally, or did they glance at some options while paying more attention to others? Even a single VE-based spatial-thinking item like this could give rise to multiple data points that could be captured and used to calculate a candidate's score.

If we then move from item-based assessment to task-based assessment then the scope for capturing even more data points increases further. Take the example of an office-based simulation or inbox/e-tray. The candidate may have complete freedom to choose the order of e-mails they respond to in the simulation, how they plan their schedule, or which of the virtual characters they choose to interact with.

So as VE-based simulations impose fewer constraints upon the candidates' choices (compared to traditional multiple-choice tests), they can capture many more metrics about candidates' behavior.

With all these potential data points and variables, how then do simulation designers know which are relevant and which are spurious? After all, with the ability to capture so many variables in VE-based simulations, there is a likelihood that some of them will significantly correlate with future job performance, even though the correlation may be spurious and ultimately indefensible in a selection context.

There are methods that can help simulation designers focus on meaningful data arising from candidates' behavior in virtual worlds, and two examples of those

methods are discussed here. The first is to develop a priori hypotheses about the relationship between candidates' VE behavior and real-world behavior. In the example of the spatial-thinking item mentioned previously, we might hypothesize that response time might be negatively correlated with spatial ability: candidates with high levels of spatial ability will answer the question more quickly. A second hypothesis might be that the extent to which the candidate rotated the three-dimensional object before choosing a response is also negatively correlated with spatial ability: candidates with high levels of spatial ability will need to rotate the object less before choosing a response. These hypotheses can then be tested by validating virtual-world performance against real-world performance measured by, for example, a "traditional" measure of spatial reasoning or by job performance. By developing a priori hypotheses in this way, simulation designers lessen the risk that candidates' suitability for a job will be measured by VE-based metrics that prove to be only spuriously correlated with real-world behavior.

A second approach is to treat the virtual world in the same way as physical assessment centers and use human assessors to observe and score candidates by using behaviorally anchored rating scales. For example, candidates participating in a VE-based simulation could be observed by assessors who are "present" in the VE, but invisible to candidates. Some VE platforms such as Second Life and SAIC's OLIVE allow for the simulation to be recorded and played back, not only as a movie but also as an immersive three-dimensional replay of the simulation, allowing assessors to repeatedly view the candidates' behavior, each time from multiple vantage points.

4.3.4 Knowing Your Audience

At the beginning of this section, deployment was described as one of the first factors that are taken into consideration when developing a simulation. Therefore, designers must very early on seek to understand the needs of their audience, the candidates. This information will inform not only the design of the simulation, but also the choice of technologies used to deliver that simulation.

Any information appearing in print, including this chapter, is soon surpassed by the rapidly evolving technology that drives the Internet. Therefore, simulation designers should seek the most current information available about their candidates and the candidates' technology—browser, plug-ins, operating systems, and so on. Two sources of information are StatOwl and W3Techs.

4.4 Conclusion

Deploying simulations for assessment over the Internet presents simulation creators with a series of challenges. However, as other chapters in this book demonstrate, there are compelling reasons for their use. The first step in designing a simulation

is to understand the constraints imposed by this medium of assessment. Once these constraints are understood and accommodated, simulation designers can make extensive use of a range of tools and technologies to deliver engaging and valid assessment simulations.

4.5 Resources

This is not an exhaustive list, but is intended to help readers locate many of the resources mentioned in this chapter.

4.5.1 Authoring and Deployment

- Adobe Flash: http://www.adobe.com
- Articulate Storyline: http://www.articulate.com
- ClicFlic: http://www.clicflic.com
- HTML5: http://dev.w3.org/html5/spec/single-page.html
- Microsoft Silverlight: http://www.microsoft.com/silverlight
- Oracle Java: http://www.java.com

4.5.2 Character Animation

- 3ds Max: http://usa.autodesk.com/3ds-max/
- Blender: http://www.blender.org
- CodeBaby: http://www.codebaby.com
- Digital Video Toonz: http://www.toonz.com
- Moviestorm http://www.moviestorm.co.uk
- Muvizu: http://www.muvizu.com
- Reallusion Crazytalk: http://www.reallusion.com/crazytalk
- Reallusion iClone: http://www.reallusion.com/iclone
- RETAS: http://www.celsys.co.jp/en/products/retas/index.html
- Smith Micro Anime Studio: http://anime.smithmicro.com
- Toon Boom: http://www.toonboom.com
- Xtranormal: http://www.xtranormal.com

4.5.3 Text to Speech (TTS)

- Cereproc: http://www.cereproc.com
- Loquendo: http://www.loquendo.com

4.5.4 Accessibility

- Section 508 Guidance: http://www.section508.gov
- Web Accessibility Initiative: http://www.w3.org/WAI
- World Wide Web Consortium: http://w3.org

4.5.5 User Statistics

- StatOwl: http://www.statowl.com
- W3Techs: http://w3techs.com

References

Adobe Systems Incorporated. (2011). Flash to focus on PC browsing and mobile apps. Adobe featured blogs. http://blogs.adobe.com/conversations/2011/11/flash-focus.html. Accessed 7 Dec 2012.

Adobe Systems Incorporated. (2012). Statistics: Penetration by version. http://www.adobe.com/uk/products/flashplatformruntimes/statistics.displayTab3.edu.html. Accessed 7 Dec 2012.

Ausburn, L. J. (2012). Learner characteristics and performance in a first-person online desktop virtual environment. *International Journal of Online Pedagogy and Course Design, 2,* 11–24.

Brandt, A. (2012). Time to give Java the boot? PCWorld. http://www.pcworld.com/article/261843/time_to_give_java_the_boot_.html. Accessed 7 Dec 2012.

Deep Blue Sky. (2012). HTML5 and CSS3 support. FindmebyIP.com. http://fmbip.com/litmus/#html5-web-applications. Accessed 7 Dec 2012.

Draney, J. (2012). HTML5 and the future of online games—SitePoint. SitePoint. http://www.sitepoint.com/html5-and-the-future-of-online-games/. Accessed 7 Dec 2012.

Fox, J. (2010). What does a corporate web video cost? 25 Factors (with prices) that affect corporate video production costs (Web log post). http://onemarketmedia.com/blog/2010/03/what-does-a-web-video-cost-25-factors-with-prices-that-affect-video-production-costs/. Accessed 14 Dec 2012.

Franco, A. (2010). Mass confusion: The hysteria over Flash, Silverlight, HTML 5, Java FX, and Objective C. Anthony's Blog. http://anthonyfranco.wordpress.com/2010/02/01/mass-confusion-the-hysteria-over-flash-silverlight-html-5-java-fx-and-objective-c/. Accessed 7 Dec 2012.

Hawkes, B. (2012). Review of animation agencies rates. Unpublished raw data.

IMDb.com. (2012). Up (2009). http://www.imdb.com/title/tt1049413/. Accessed 7 Dec 2012.

Jobs, S. (2010). Thoughts on Flash. Apple.com. http://www.apple.com/hotnews/thoughts-on-flash/. Accessed 10 Dec 2012.

Le, Grecs. (2012). Java, Flash, and the choice of usability over security. Infosec Island. http://www.infosecisland.com/blogview/22381-Java-Flash-and-the-Choice-of-Usability-Over-Security.html. Accessed 7 Dec 2012.

Linden Research, Inc. (2012). History of Second Life. Second Life Wiki. http://wiki.secondlife.com/wiki/History_of_Second_Life. Accessed 7 Dec 2012.

Macromedia. (2002). Macromedia Flash MX—A next generation rich client. Macromedia. http://download.macromedia.com/pub/flash/whitepapers/richclient.pdf. Accessed 7 Dec 2012.

Mastrangelo, T. (2012). North American telcos struggle to gain broadband subscriber continues in 3Q12. The voice of broadband. http://broadbandtrends.com/blog1/2012/11/12/north-american-telcos-struggle-to-gain-broadband-subscriber-continues-in-3q12/. Accessed 7 Dec 2012.

Maxion, R. A., & Reeder, R. W. (2005). Improving user-interface dependability through mitigation of human error. *International Journal of Human-Computer Studies, 63,* 25–50.

Mennecke, B., Hassall, L. M., & Triplett, J. (2008). The mean business of second Life: Teaching entrepreneurship, technology and e-commerce in immersive environments. *Journal of Online Learning and Teaching*. http://jolt.merlot.org/vol4no3/hassall_0908.htm. Accessed 4 Dec 2012.

Richardson, A. E., Powers, M. E., & Bousquet, L. G. (2011). Video game experience predicts virtual, but not real navigation performance. *Computers in Human Behavior, 27,* 552–560.

Satalich, G. A. (1995). Navigation and way finding in virtual reality: Finding the proper tools and cues to enhance navigational awareness. Unpublished Master's thesis, University of Washington, Seattle, USA.

United States Government. (2011). Section 508 home. Section 508. http://www.section508.gov/. Accessed 7 Dec 2012.

W3C. (2012). Web Accessibility Initiative (WAI). World Wide Web Consortium (W3C). http://www.w3.org/WAI/. Accessed 7 Dec 2012.

Weiss, C. (2012). Bridge over troubled authoring tool waters. E-Learning 24/7 Blog. http://elearninfo247.com/2012/11/13/bridge-over-troubled-authoring-tool-waters/. Accessed 7 Dec 2012.

Winokur, D. (2011). Flash to focus on PC browsing and mobile apps; Adobe to more aggressively contribute to HTML5 (Web log post). http://blogs.adobe.com/conversations/2011/11/flash-focus.html. Accessed 9 Dec 2012.

Zickuhr, K., & Smith, A. (2012). Digital differences. Pew Internet & American Life Project website: http://pewinternet.org/~/media//Files/Reports/2012/PIP_Digital_differences_041312.pdf. Accessed 10 Dec 2012.