Chapter 7
Future Directions in Games for Serious Contexts: A Conversation About Transferability

Vanissa Wanick, James Stallwood, and Guilherme Xavier

Abstract This chapter provides a conversation in the form of an opinion piece about strategies commonly utilized in games that can be "transferred" to Serious Games (SGs) and games for serious contexts. The aim of this chapter is to provide different perspectives and examples that are currently utilized by entertainment games that could be utilized in SG development. SGs are often developed for particular situations, and with that, the development process might be attached to specific stakeholders, becoming, most of the time, a "one-off" product, which may limit the SG life cycle and game repurposing. This chapter brings with three complementary perspectives to address future challenges and opportunities regarding emerging aspects of player agency and SG modification and transferability across different contexts. First, we discuss emergent possibilities, bringing examples from digital entertainment transferability. Second, we take into consideration "modding" strategies to provide insights for SG modification and transferability, discussing the role of the "context" in games development. Third, we demonstrate the importance of AI emotion modelling to inform better game design. To conclude, we respond to these ideas and provide suggestions for SG research and practice.

Keywords Personality vectors · Transferability · Serious games · Position paper · Modding · Emotional modelling

7.1 Introduction

Serious games (SGs) and gamified applications utilized in non-entertainment contexts have the potential to promote positive behavior but also keep the user

V. Wanick (✉) · J. Stallwood
University of Southampton, Southampton, UK
e-mail: vwv1n12@soton.ac.uk; J.E.Stallwood@soton.ac.uk

G. Xavier
Pontifical Catholic University of Rio de Janeiro, Rio de Janeiro, Brazil
e-mail: guix@puc-rio.br

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2023

K. M. L. Cooper, A. Bucchiarone (eds.), *Software Engineering for Games in Serious Contexts*, https://doi.org/10.1007/978-3-031-33338-5_7

engaged in a specific activity. Due to their attachment to the "context," these games and applications may become a "one-off" product, being difficult to replicate outside their own environment. But what if the process of creating and developing serious games could be fed by other models, imported both from the entertainment area and from constituent approaches of automatism programming techniques, mediated by the experience of their designers?

This chapter provides a conversation and an outline of topics usually utilized in games that can be "transferred" to serious games and games for serious contexts. This approach we have called as "transferability." The concept of transferability discussed in this chapter can be twofold. From one side, transferability is about the generalization of "solutions" and from another, transferability accounts for the extent to which a solution can be effectively achieved in another context. Thus, considering that serious games are designed based on and for a particular context, we believe that by discussing transferability, we can provide recommendations for improvement and insights to design and develop more effective SGs.

In research, transferability relates to the degree to which that research method or output can be transferred to other contexts. Transferability might be at times combined with generalization, which means that the output can be applied into other contexts. For SG design, it means that these games are designed with a purpose in mind; however, would that mean that a serious game can be only applied to one single context? As [1] mention, the definition of the term SG is related to cultural practices, but as these practices evolve, the range of the term might also change. A SG can be used to communicate a message, enable training/cognitive or physical capabilities, and facilitate data sharing and collection (e.g., crowdsourcing and citizen science applications, such as *Foldit*) [1]. This means that SGs have a clear purpose, though it does not imply—not directly—that the game itself needs to be designed for that single purpose.

Yet, the term "serious games" deals with a conflict between what is admitted as purposeful seriousness and what is expected from a game as an activity aimed at the enjoyment of its participants. If the game exists at play time, this is where, at the "Game," we will drop our anchors to take a look around. In addition to this discussion, therefore, we will first consider the "game" part of the term "serious game," implying in the dialogue that the difficulty for designers in making their projects engaging is not in the content but in the way in which it is presented for interactive participation.

Considering this, this chapter discusses transferability, from the perspective of the authors, in three sections: (1) transferability of competences and skills constituted by gaming practices (e.g., the idea of games as instruments) and transferability via games as tools (e.g., *modding* practices), (2) transferability of "informal" game development practices via Game Jams and games developed as a commentary about a particular "serious" context (e.g., health issues), and (3) personality vectors as an AI mechanism to improve the quality of in-game agents. Each section has a particular treatment given by the authors, with relevant examples. We believe these examples extend debates in the literature around repurposing of SGs [2, 3] and agent-oriented software engineering [4].

Despite the challenges of developing games for serious contexts (e.g., user engagement, costs of production, metrics of the effectiveness of the game/application, etc.), software engineers and designers have been working toward personalized strategies and algorithms that provide users with unique experiences [5]. Artificial intelligence (AI) might be able to solve a few issues with personalization; however, gameplay adaptability might be difficult to achieve when the game is already made.

Games applied in serious contexts might also depend on stakeholders' ability to deploy the game in that context (e.g., by recommending and reading results of a health application or designing experiences for (and with) students). Thus, when designing games for serious contexts and gamified applications, it is important to understand how much control would be given to its "users," how and when, in order to make it relevant.

This chapter aims to address future challenges and opportunities regarding emerging aspects of player agency and SG modification across different contexts through a set of conversations posed by the authors. To conclude, we respond to these ideas and provide suggestions for SG development. For software engineers, we discuss the potential of modular approaches, together with the application of personality vectors to enrich intelligent tutoring mechanisms and potential practices for the early stages of SG development.

7.2 Serious Games Design Transferability: Setting Up the Conversation About Purpose, Tools, and Instruments

In this section, we discuss aspects that blur the line between entertainment and serious contexts, such as games designed as a commentary about a "serious" context, but that are not SGs per se as their main purpose is entertainment. We will use two examples to discuss the concept of transferability in the design of two games: *Before I forget* and *Hellblade: Senua's Sacrifice*.

Before I forget (Fig. 7.1) tells the story of a woman trying to reconnect with her past, but it creates several scenarios that reflect the emotional portrait of dementia. In an article published by *The Guardian* [6], the authors mentioned the idea emerged from a Game Jam that had in fact a serious context to tackle. Game Jams (GJs), from Global Game Jam (GGJ) to many others (particularly hosted in the platform *itch.io*), can support a new landscape for game design and development, since it provides a safe space for people to collaborate and create together in an informal setting. The reason for that is that these platforms and events can offer innovative and interesting new mechanics since it allows quick and rapid prototyping games. GJs are usually thematic and have an initial point of interest. Themes may vary, from indigenous communities to keywords like "roots" (GGJ 2023s theme). Within these themes, questions might emerge; for example, what is the best way to represent the concept of "roots?" What do "roots" represent? How might these ideas become interesting mechanics? These are common questions that emerge in a brainstorming session,

Fig. 7.1 *Before I forget* (Source: Threefold games)

which may emerge from a problem statement [7]. But what if the questions become more technical and more subject specific?

Let's bring back *Before I forget* and discuss the questions suggested by the developers and designers. During the early stages of development, the creators mentioned that one of the questions they had was "What happens when we lose our memories?" [6] Then the creators moved to design questions, such as on how to implement and how to go further into developing the narrative and the other design components. The key aspect for the innovative factor in this case was the transferability from one discipline to the other and having two questions being asked at the same time: one related to research and scientific aspects and the other related to design and development (on how to make the final prototype). *Before I forget* is not a SG but has serious contexts involved in its design and core experience.

The same can be said about *Hellblade: Senua's Sacrifice*, which tackles psychosis. For the design of this particular game, designers worked together with professors and neuroscientists, and the research was fed into game development via the description of hallucinations and other experiences [8]. The game itself portrays a *Viking* environment, but it is the experience that conveys the psychotic state of the main character (*Senua*). *Senua* is a traumatized Celtic warrior on a quest to the *Viking* underworld, Hel. Thus, there is a layer of "fantasy" being added to a "serious" context. It is worth mentioning that the game, however, has not emerged from a GJ, as compared to *Before I forget*, but from a research project.

Both *Before I forget* and *Hellblade: Senua's Sacrifice* had their base on scientific and subject specific knowledge but are entertainment games. However, this does not mean that these games cannot be used in an educational setting. In fact, they might be great tools to teach about particular health conditions.

In this sense, it is important to differ two concepts that are fundamental when considering games, and serious games in particular, with a focus on their purpose and transferability: games as "tools" and games as "instruments." Although they seem like only terminological differences, they evoke different interpretations and thus different perceptions of purpose.

When we consider games as "tools," we are assuming their functional characteristics as something participated to resolve a problem. It should be noted that a game as "problem-solving activity, approached with a playful attitude" [7] is a sufficient ontological definition of game, in the absence of a definitive one, that both the area and the related disciplines have not yet been able to consolidate. That said, when we think of games as "tools," we are assuming that they fulfil a dual purpose: the first within a diegetic dimension and in compliance with rules to reach an objective and second in a sphere that expands in the experience of its own practice.

For example, there is a recent appeal in translating work activities that would otherwise be considered completely devoid of fun, bringing to make-believe the "responsibilities" of a simulated job-like activity. We highlight here the games *Plane Mechanic Simulator* (Disaster Studio/Cobble Games, 2019), *Euro Truck Simulator 2* (SCS Software, 2012), and *PC Building Simulator* (The Irregular Corporation, 2018), which somewhat could be perceived as serious games in convenient contexts (if the player is a mechanic, trucker, or computer technician), beyond those they are "originally" related to.

In the first game, *Plane Mechanic Simulator*, the player is invited to consider that World War II airplanes are like an assembly of puzzle pieces that occupy a specific location at a specific time, which we can summarize as a *space-time-problem*, in the mechanical structure of a complex vehicle. For there to be a repair (pointed out bureaucratically, on a clipboard), the player must navigate a three-dimensional structure of the object to be mended, laboriously removing screws, plates, and exquisitely detailed parts to meet the objective of the demand, later carrying out the reverse process of disassembly to obtain victory. What at first glance may seem like an activity worthy of Sisyphus, it actually finds an echo in a mechanistic and structuralist society, the one that seeks to reduce the distance between *modus faciendi* and *modus operandi* of cultural assets in the virtual world. The industrial complexity has as a summary a kind of *gap* between what is made and what is used, which in part explains the large number of different "maker kits" now on online sales to satisfy hands that are not busy with manual duties but overloaded with intellectual work, as a way of aesthetic "compensation." Thus, if there is a low possibility of actual building nowadays in our lives, at least the experience of *virtual* "building" is possible, seductive, and free from consequences.

In the second game, *Euro Truck Simulator 2*, the player takes the wheel of a truck to transport goods, of all kinds, along the roads of Europe. There is a version located in the United States by the company itself, but this *spin-off* title little or nothing differs in the result of a driving experience, since the community also manages to create other *mods* to update the game to other parts of the world and their respective highways. If the player has no interest in trucks, but in buses or even cars, it is too possible for unofficial installations to adapt the game to the taste of its driver, which

is seen as normal behavior by the developer company. This, in part, solves technical issues (in the "eternal pursuit of realism as an end") and in part interferes with the way the game is consumed (beyond the usual player and in more *spectacular* ways). On the *Twitch* video platform, specialized in broadcasting matches from this and many other games, there is vast content of "drivers" getting involved in hypothetical scenarios of dangerousness on unrealistic roads, in a kind of "self-assured skill tests," which in the game is a concern but not the most important. At least, no more than following the traffic rules (whose fines are immediately charged to the driver), exploring kilometers and kilometers of territories based on typical old continent environments without running out of fuel or passing out from sleep deprivation, and delivering packages on time for XP acquisition.

In the third game, *PC Building Simulator*, the player takes over a private shop specializing in repairs and installation of computer parts, retracing a path similar to *Plane Mechanic Simulator* in a genre that we could call *adminpuzzle:* the traditional puzzle of digital games plus financial responsibilities such as those of *Euro Truck Simulator 2* and their administrative schedules, garages, and employees. The metalanguage of this game is too provocative and does not allow us to exempt a comment about a computer game in which computers are based on "real" computers, and even their parts are sponsored by "real" companies, dedicated to computer construction that must be built with performance for digital games. The process is cyclical and, therefore, meditative: tasks need to be carried out on different equipment, seeking with audio-visual assets to "ludify" something that would otherwise be perceived as just a technical work devoid of "epicity," common in other interactive digital works.

As well as these three games, many others seek, in the rigor (or lack of) of the simulations, to bring their players the feeling of a metrical tooling operation. There are learning curves and, therefore, results that demand dedication for a certain number of hours. The game is thus a "tool" to be used to satisfy an estimated result, generally far from the reality of its player. As games are opportunities for other experiences, there is meaning in challenges that deal with this alternative and, therefore, experimental prerogative.

Minecraft, for instance, can be said to be a "simulator" that allows players to experiment with. The "game" has playful and gameful components, allowing players to explore an open world but to also play with rules (surviving mode). *Minecraft* has many versions including an educational version, which then becomes a "serious toy" (using [1]'s terminology). This "version" and its *mods* are accompanied by teaching resources. In this case, the "real" context (e.g., classroom) is still imperative for the learner to achieve the learning objectives. What makes *Minecraft* an interesting example about transferability is that the game can be applied into other contexts, which comes together with the idea about repurposing a game for a particular context. Can a SG designed for that purpose be repurposed and have "transferable" components? So, could we do the reverse? *Foldit* is potentially a good example (see Fig. 7.2). When *Foldit* was launched in 2008, its first challenge was to decode proteins, which then were applied to solve issues related to the COVID-19 global pandemic (see Fig. 7.2). *Foldit* did not change its core and still remained

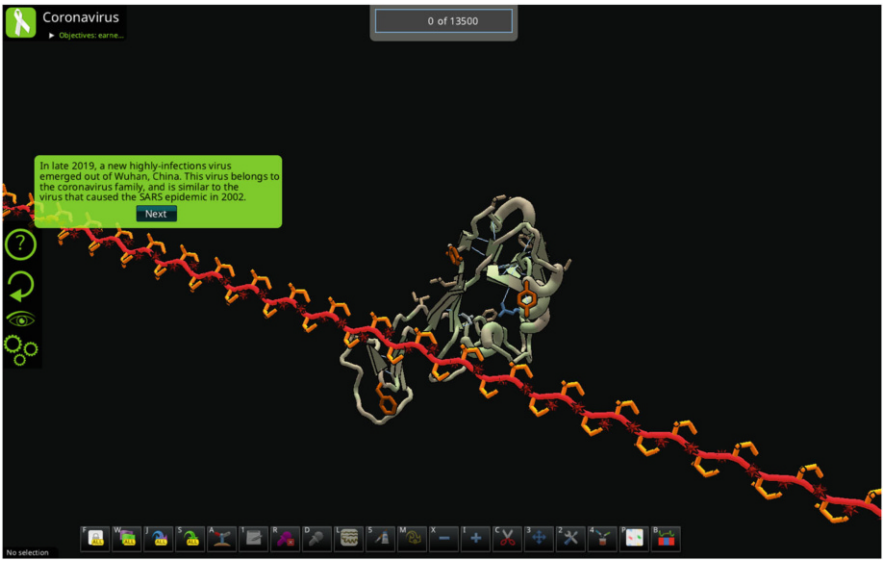

Fig. 7.2 *Foldit*: COVID-19 version (Source: Screenshot from authors based on http://www.fold.it)

with the same mechanics, but it changed its challenge and potentially the way these were presented to players. Therefore, would "transferability" mean that the core mechanics stay intact? Would that not be the same as reskinning an existing game?

Jenkins et al. [9] proposed that educational game design requires negotiation of identities, assumptions and meaning, as games it might not be clear if the games are seen as just entertainment or just educational pieces. This might be a matter of perspective and purpose, being the goal of the game to support learning objectives or particular goals, and the "fun" is the way to achieve that. Nearly 20 years later, there is still a need to negotiate "identities," which [9] represent by teachers, designers, students, and so on. Nearly 20 years later, these roles and identities are revisited with the rise of the *modding* culture, like those common in platform/gamehub *Roblox* (Roblox Corporation, 2006). Modifications have been used in design and in games for serious contexts, as mentioned before. However, these were still modifications mostly made by teachers/designers and less from the perspective of players/students. In fact, one of the most challenging aspects is to be able to identify which games are "moddable" [10].

We talk now about games as "instruments," that is, as an activity that demands time invested in skill, constituted by dedication and some *tuning*: the operational adjustments that are necessary so that the result is in line with what is expected for a driven sensation to fulfil rule-imposed objectives. In these games, victory is not enough but the certainty of "being one with the game" in the fullness of knowing how to know. Thus, the more used the instrument, the better the apparent and internalized result, which we can understand as a learning process.

Csikszentmihalyi's [11] *Flow* is immediately expected to appear right here: by becoming one with the game, the player awaits the trance that allows him to advance and advance, seeking in virtuosity, both personal and social satisfaction.

The transferability of competences and skills constituted by gaming practices has been the core of serious games since their first applications for school renewal and training in the 1970s of one or more proficiencies; it becomes common with the expansion of digital electronic accessibility in the 1990s and with the multi-mediatization of cybernetic microcomputing in the 2000s. Although they belong to another dimension of interactivity with a purpose (which would go against the usual expectation of a game for its "promising empty" condition precisely because it belongs to the sphere of interstitial activities of modern and productive societies). While "instruments", games require their participants to understand all the resources involved in order to take better advantage of them. In addition to their mechanics, dynamics, and aesthetics [12], games as an activity exist only ongoing, but as an experience, exist both ongoing and after, in the form of affective memory and tacit knowledge.

Whether "tools" or "instruments," games can be "recommissioned" for other practical purposes as long as they are not deprived of their main conditional characteristics: fantasy and control. This means that entertainment games can be appropriated, via an exercise of "metaphorization"; that is, the game has a symbolic meaning that could function as a metaphor or analogy. After all, as a means of communication and expression, games assume themselves as promoters of a purpose. And being recognized as such, they facilitate the migration of use knowledge from one context to another. As a subversive sample of this kind of entertainment gaming transferability we are talking about, let's discuss the *modus operandi*, the use of the keyboard for first-person shooter games.

Back in the year 1997, *Quake* became an outstanding commercial success by iD Software. Until the emergence of complex games that require the simultaneous use of mouse and keyboard, video games had a certain amount of possible and allowed inputs, establishing two paradigms sufficient for interaction, the "lever" (generally pushed in the direction of the command) and the "button pressed" (to activate or deactivate an action). The use of the keyboard had in the function keys F1 to F12, Enter, Space Bar, and directional arrows an industrial agreement of use; therefore, they were foreseen in function of the programs that made use of their positions and functions. When that year Dennis "Thresh" Fong beat Tom "Entropy" Kimzey in the first national *Quake* tournament, he went on to popularize an inadvertent use of W, A, S, and D keys as the new "directional" keys. Anyway, the next first-person shooter games like *Half-Life*, *Counter-Strike*, and *Unreal Tournament*, to name a few popular examples, gave up the directional arrows due to this new "imported" adaptation scheme, which is still considered *default* even for games that are not first-person shooters.

Speaking of the present, in search of a future in which SG invites the versatility of its samples to acquire welcome *flow* states, it is necessary to investigate opportunities in the production of game assets and its relation with its players. Transferability, then, becomes a token both serious and entertainment games must rely on, focusing

on engagement behavior, directly associated with emotional responses. Next, we bring some technical suggestions about that.

7.3 Going Technical: Personality Vectors as a Strategy Toward Emotional AI Modelling

The aim in this section is to first set a premise for a concept of personality vectors and then show with an extended example how these personality vectors might be used in a more complex example. This section deliberately discusses examples outside the SG field, particularly looking at a potential strategy to evoke emotional behaviors of non-player characters (NPCs). The application of these examples is addressed in the conclusion.

To begin, let us think of a guard patrolling an environment. This guard will patrol until an enemy, the player, is spotted and will then engage the enemy in combat or flee from the enemy dependent on if the guard is sufficiently healthy and armed to do so. We will assume that the guard begins their duty fully in good health and well-armed for the purpose.

Regardless of whether we might use a state-machine or a behavior tree or any other kind of model for this NPC, the design is similar for all of them. The guard receives information from the game-world and its own current situation and will decide upon certain actions if pre-conditional statements are met. For example, suppose that it is decided that in order to engage an enemy, the guard must first be able to see the enemy, be reasonably healthy (e.g., health > 50%), and have an abundance of ammunition for a number of shots to be made (e.g., 10 shots). If those conditions are met, then the reasoning framework for our guard will make them engage their foe. This collection of preconditional statements and actions can be called the NPC's "strategy" or "strategy framework." One could think of it as the training manual that the guard received before they took the job.

This strategy framework may be a sufficient basis for all guards, but the way it works opens an obvious in-game question: how does the guard know how healthy they are? So, we might abstract this further and ask what does this guard have the right to know accurately about themselves from moment to moment? As the preconditional statements in the strategy framework rely on the data collected by the guard, why should that data be processed directly rather than being put through some data judgement protocol first?

If the guard is wrong about spotting the player, then the guard may open fire into a harmless jacket hanging on a door. If the guard is wrong about being sufficiently healthy to take on the threat, they will find themselves at quite a disadvantage. If the guard is wrong about having enough ammunition, they will quickly find themselves in an action movie cliché.

For simplicity's sake, we could imagine the data judgement protocol to look something like this and using a simple probability:

```
NPC_Judgment():
r = random_number(0, 1)
IF r <= 0.05
   THEN RETURN random_number(-10, 10)
ELSE RETURN 0
```

For the guard's preconditional variables (health, ammunition, enemy_spotted), there would be two distinct variables used: the actual, or true, value variables and the perceived variables where the returned value of the NPC_Judgment protocol is added to the actual value. The result is that instead of using the actual values in the strategic framework's conditional reasoning, the perceived values are used instead.

This would provide greater nuance and difference to the guard's behavior at relatively little cost and, more importantly, without having to alter the basic strategy framework at all. However, we could go further. Though we have chosen arbitrary numbers for the probability and the ranges to add for the perceived variables, these could instead be linked to a predesigned personality for the guard or change over time according to circumstances. We can rewrite parts of our NPC_Judgment protocol to account for these options.

```
misjudge_rate = 0.05
misjudge_extreme = 10
NPC_Judgment():
r = random_number(0, 1)
IF r <= misjudge_rate
 THEN RETURN random_number(-misjudge_extreme, misjudge_extreme)
```

In predesigning the personality for the guard, we might decide to make a guard that is more prone to making misjudgments, and so we would increase the misjudge_rate to be a higher value and more likely to occur. Or we might decide that our guard character is even more inaccurate with their perceptions than an ordinary guard and increase the misjudge_extreme value. Or, indeed, we could decrease those values for more seasoned guards and any mixture in between. In taking this approach, we allow ourselves to make personality and experience changes to our guard agents without changing the strategy framework for all guards. Divorcing this process enables us to safely experiment with these parameters without the need for tedious minute work in the strategy framework itself.

We might decide instead to fix these values to some external *stimuli* as well. For example, consider the type of guard who on seeing half of their comrades. If instead we took something like the number of times the guard has seen a fallen comrade and either mapped that or used it as a factor for the guard's misjudge_rate and misjudge_extreme values, then we begin to get something approaching a basic fear index. When people are scared, they make mistakes. Our NPC_Judgment variables might instead look a little like this:

```
misjudge_rate = 0.05 * fear_value
misjudge_extreme = map(10, 30, fear_min, fear_max, fear_value)
```

where in the first instance the fear index acts as a factor for the misjudgment probability and in the second instance the misjudgment extreme is mapped between 10 and 30 depending on the current value of the fear index compared with some maximum.

If we were to have multiple personality indices, we might weight their effects on these judgment variables differently according to how we thought they might apply or by personality design. Suppose our guard has two personality indices: fear and anger. We could design a guard that is affected more by one than another or by both equally. So, a guard who is more cowardly might be made like this, misjudge_rate = (0.1 * fear_value) + (0.02 * anger_value), whereas a guard that is more easily caught in the red mists of rage might be made like this: misjudge_rate = (0.01 * fear_value) + (0.2 * anger_value).

In either case, we have a liberty to model our guards according to personality in whichever way we choose. A fearless guard might take no influence from fear at all.

In the event we have multiple personality indices to track and to factor into our NPC_Judgement protocol, we could create a personality matrix where one row, say the first, is the effect vector for the guard's personality and the second row are the personality indices generated from external events in the game. Treating the rows as single vectors, our simple model for the misjudge_rate variable can be calculated with the dot product of these two rows.

Of course, we don't have to use the dot product approach or the mapping approach described above; these are simple implementations. However, going forward with this idea, it is worth remembering our two key ideas: a strategy framework which does not alter at all between different agents and a personality matrix or vector of indices which is used to add a variation to the behaviors described in the strategy framework.

Let us now move on to a different kind of problem: NPC poker players or poker agents. Poker, and in this case referred specifically to Texas Hold 'Em, as with many other card games, has standard conventions that accompany its rules, and because lying is an intrinsic part of playing poker, those conventions make the game both stable and unstable in trying to determine the actions of a player.

For example, it is a common convention that the closer a player is to the dealer chip from a clockwise position at the table, the stronger their hand should be if they choose to call or raise a bid. The reasoning for this strategy is simple; players closer to the dealer chip must act before those further away, and it is better to be in a position of strength. Similarly, if a player has the dealer chip or is close to the dealer chip on the right, they can play weaker hands knowing that they do not have to act before anyone else.

If we were to design a poker agent to use this strategy alone in a limited experiment of opening plays, we would be missing a fundamental part of the game. Our agent would consider its position, analyze the strength of its hand, and make some judgment, probably probabilistic on whether it should call, raise, or fold. However, *what we are missing in this implementation is the ability to lie*. Because of the convention that is held due to the collective experience and knowledge of poker players down through the ages, a big blind who raises represents a strong or monster

hand. At this point, we should carefully not the poker terminology "represents." We do not say the big blind player has a strong hand, only that they represent one and act as if they do.

The curiousness of poker is that it is a game seeped in personality. As a player, we must assume our competitors are trying to be truthful about their hand and intentions even though there is an excellent chance that they are lying to us.

Therefore, when we design such agents, we must be mindful of many more factors. We could, for example, begin with a simple protocol for analyzing the strength of a hand. There is, for example, an approximately 6% ($\frac{\binom{13}{1}\binom{4}{2}}{\binom{52}{2}}$) chance of being dealt a pocket pair. In the pre-flop portion of the game, this is a major strength. Afterall, there are only 13 values of pairs, and their relative unlikelihood means that at a table of five people, if you have a pocket pair at this stage of the game and cards were turned, you'd have a very good chance of winning. By the end of the game, five cards, the likelihood of other players also having a pair increases significantly as do the chances of your pair being beaten.

What is useful for designers is that poker is a game which illustrates the folly of holding onto a strategy framework only model. Our goal is not in creating poker agents that win optimally wherever they can but ones that *act like poker players to increase the verisimilitude of the game experience.*

This then leads us to the question, what of the personality vectors/matrices? How do we use them? We must first ask ourselves what is a reasonable area in which a poker player could make a mistake in their reasoning? What is the equivalent of misjudging one's ammunition or health status in poker? Finally, because of the nature of the game itself, how can we abstract misjudgment protocols to include intentional misrepresentation of our hand, bluffing?

Let's start with some assumptions about our poker agent. Firstly, we will assume that our agent will not forget the cards they have. While this does occasionally happen, the agent is technically free to look at their cards at any time so we will not consider this to be a valid misjudgment opportunity. Secondly, we will assume that our agent knows the rules of the game fully and isn't playing different variants unknowingly. Thirdly, we will assume our agent can perform simple arithmetic operations like percentages of the pot, cards seen versus cards remaining, etc.

As there are many factors that could be in play in a poker game, we will only use one of them: hand potential. Using only the values for the two cards in the agent's hand and knowing if they are suited (sharing the same suit) or not ("off suit" or "off"), there are 169 possible hands an agent might have. We can value our hands numerically from the best possible two-card combination (two aces) to the worst possible two-card combination (7 – 2 off). As with our earlier guard example, it should be possible to assign a misjudgment for card strength in the same way we discussed the idea of health or ammunition earlier.

Card strength is not only relative to other cards but also indicates an idea of playable freedom depending on where the agent is sat relative to the dealer chip. In other words, a hand's strength is not dependent only on the values of the cards but

also on where the player is sat in combination. If there were six people at a table playing the game, then the best position would generally be in the sixth chair. If we label each seat, beginning with the small blind at 1, then the seat position for the player can be used as a starting point. If our agent were sat next to the button, then they would have a position score of 5.

To begin to formulate a score, let's call this variable hand_potential. Thus, so far:

```
hand_potential = position
```

To keep things simple, we could assign each card combination a score using 1 (worst) to 169 (best). Combining these terms into our hand_potential variable, we get:

```
hand_potential = position(hand_strength)
```

If we use the extremes for our range of possibilities, this means that we have two cases of comparison to judge our combination. The first is that at our six-person poker table, the worst possible hand in the best possible position is comparable to the sixth worst hand in the worst position. The second is that our 28th worst hand in the best possible position is comparable to a pair of aces in the small blind position. Clearly this simple combination doesn't really account for hand strength appropriately. We can fix this by either reducing the effect of the position on the final score or increasing the effect of the hand strength.

If we square the hand strength instead, we get a more satisfactory result. The worst hand in the best position is only better than the second worst hand in the small blind position. Whereas with this model, the best hand in the worst position has a worst score than the 70th worst hand in the best position, a little shy of half all-possible hands. If this is not satisfactory still, and this is a very simple modelling for the purposes of illustration, we could add some factor before we square the result. We might add ten to the pre-squared hand strength if it is a pocket pair and five to the suited cards to add a reasonable distinction. A small pair (generally considered to be less than ten) has more chance of becoming a three of a kind than suited cards have of becoming a flush or a straight.

```
hand_potential = position((hand_strength + (suited? + pair?))^2)
```

This model for hand potential may not be perfectly nuanced, but it is hopefully sufficient for a basic agent to help make its judgments about play decisions and basic enough to implement simply. Indeed, we might calculate hand strength according to the number of over cards as well, but we'll leave that distinction out for the moment. As we now have a judgment variable, we can apply similar principles of alteration as with the guard assessing their health and ammunition.

In this instance, we would see agents making raises when they should call or folding when they should call, etc. These changes would form a slight variance to the accepted strategies replicating what we see real poker players do, often to the chagrin of championship bracelet winners around the globe who take a dim view to such plays.

This leads us to the first of our potential indices for the personality vector: *experience*. Experienced players will more readily compute the conventions of play because it helps them size up their opposition. Experience will lead a player to better evaluate their hand and position in the game and cause them to act more accordingly with the conventions as they are tried and true. Inexperienced players, who maybe are less aware of their position and the conventions are more likely to make risky and daring moves if they move at all.

Likewise, with experience, a player might be described as a tight or loose player (which links to the concept of bluffing). A tight player is someone who works more with the value of their hand and position and acts strictly accordingly (the kind of player that will deviate less from the conventions of the strategy framework). A loose player, conversely, is the opposite. A simple implementation of these kinds of personality on a factor such as hand potential would be to observe if a tight player does not play a hand because one or two factors are not as optimal. This being the case, a player's style as index in the personality vector would act as a limiting or gaining factor for hand potential leading loose players to make riskier plays (in effect over valuing their cards) and tight players to make much more conservative plays (in effect under valuing their cards).

Therefore, based on the two examples presented in this section (patrolling agent and poker player agent), we can expect that the strategy framework can be formulated separately from the agent's interpretation, which would be then grounded on the personality vectors. These personality vectors can vary in many ways as described, from reactions to particular *stimuli* to the ability of being able to "lie."

In the next section, we discuss the applicability of these ideas into SG development. We hope to see the adoption of more personality-based reasoning for game agents, so it will lead to a better experience all round for the players of those games. We hope also that a general paradigm model for this can be developed in much the same way other reasoning models have been created.

7.4 Conclusions

This chapter proposed a conversation about aspects that can be "transferred" from games and then can be potentially utilized in SG development. To conclude this conversation, we would like to propose several ideas on how to take these concepts into practice.

7.4.1 Games as "Tools" and Games as "Instruments"

In Sect. 7.2, we have mentioned about the transferability of competences and skills constituted by gaming practices, particularly from the perspective of games

as tools and games as instruments. In this case and in SG development, games could be repurposed if seen as a "serious toy," meaning that the game itself would have multiple facets. This could be explained via a modular approach (similar to LEGO bricks), in which the game now seen as a "toy" could evoke multiple playing settings. In this particular case, *modding* can be a methodology to be applied and investigated, particularly the aspects that make a game "moddable." Thus, by incorporating "moddable" affordances to the game, it might be that designers and developers could provide more modular approaches to development. Yet, questions still remain: How can a player learn how to modify the game? Can this be accessible to all players, or is it still for a minority of players who are familiar with development? The same can be applied for the discussion and title from [9]'s paper *You can't bring that game to school* (2003); we may, as developers and designers, actually allow students to bring the games to school and change them. Since games become even more part of today's society's culture, the choice of which game to use for pedagogical reasons could be negotiated.

Another aspect mentioned in Sect. 7.2 was games as "instruments." The transferability aspect of this category is competences and skills, which aligns with flow mechanisms, game balancing, and alignment of player-game knowledge ("know-how"). Thus, for SG development, this aspect might inform SG adaptability and controlling conventions and heuristics.

7.4.2 Early Development, Purpose, and Contextualization

In this chapter, we mentioned the benefit of GJs, particularly when developing games quickly and dynamically in teams. Serious contexts are common (e.g., sustainability, health, politics, etc.); thus, SG developers and designers might want to engage in defining small but effective game mechanics and processes in order to increase SG development quality in early stages. As noted by [13], SG effectiveness and playability tend to be evaluated in the end of product development, showing that there is still a gap for quality assurance during the early stages. We expect that perhaps via GJs (or similar participatory/co-creative events) and alignment of both research and design questions designers, developers and stakeholders can ensure SG quality but also develop innovative mechanics and design solutions together. In this case, player control and fantasy should be balanced with the learning (or behavioral) objectives, in order to inform better practices in SG development. Since playability aspects influence SG effectiveness [13], it might be that the mechanics need to be explored before aligning it with the pedagogical needs. Yet, design and research questions need to be equally balanced and addressed. Our examples in Sect. 7.2 showed how these can be addressed and perhaps this approach might help SG development in the early stages.

7.4.3 Personality Vectors, Emotional Modelling, and SG Development

Introducing personality vectors as part of the reasoning structure can add nuance and variance to the expected behaviors of non-player character agents in games in a way that increases the opportunity for unique and more lifelike behavior from those agents. What is more, the two-structure approach of keeping the personality vector separate from the strategy framework allows us to influence changes in the agent easily, cleanly, and with an overall simplicity that maintaining one structure does not allow and that does not require us to design multiple solutions for multiple agents. In essence, the strategy framework is the training manual, and the personality vector is the agent's interpretation. These ideas can be implemented toward intelligent tutors, for example, in order to generate more believability and perhaps even trust (even with an NPC being able to lie). In health, personality vectors could enhance the emotional response NPCs might have toward a sensitive topic or provide more human-like responses.

Yet, as a result of a conversation on "transferability" from general games to SG development, this chapter brings back into discussion particular questions that might be of interest and might enrich SG development and research, such as: How much of the game stems from the context? Can this be a multiplatform experience? What does it mean to design a "transferable" serious game? What can be learnt from current design processes to make these games more transferable? Can this be applied in all serious games?

We hope that there will be greater flexibility in what is understood as academic terms, since the opening is beneficial not only for the diverse realization of serious games but as an invitation to new looks at applied entertainment.

References

1. Alvarez, J., Djaouti, D., Louchart, S., Lebrun, Y., Zary, N., Lepreux, S., Kolski, C.: A formal approach to distinguish games, toys, serious games & toys, serious re-purposing & modding and simulators. IEEE Trans. Games. (2022).
2. Protopsaltis, A., Panzoli, D., Dunwell, I., de Freitas, S.: Repurposing serious games in health care education. IFMBE Proc. **29**, 963–966 (2010)
3. Chettoor Jayakrishnan, G., Banahatti, V., Lodha, S.: GOVID: repurposing serious game for enterprise COVID-19 awareness. ACM Int. Conf. Proc. Ser., 11–18 (2021)
4. Gómez-Rodríguez, A., González-Moreno, J.C., Ramos-Valcárcel, D., Vázquez-López, L.: Modeling serious games using AOSE methodologies. In: International Conference on Intelligent Systems Design and Applications, ISDA, pp. 53–58 (2011)
5. Ninaus, M., Nebel, S.: A systematic literature review of analytics for adaptivity within educational video games. Front. Educ. (Lausanne). **5**, 308 (2021)
6. Wen, A.: Before I Forget: the video game that tackles dementia | Games | The Guardian. https://www.theguardian.com/games/2018/jun/06/before-i-forget-early-onset-dementia-video-game. Accessed 17 Nov 2022
7. Schell, J.: The Art of Game Design, a Book of Lenses. (2008).

8. Lloyd, J.: How Hellblade: Senua's Sacrifice deals with psychosis | BBC Science Focus Magazine. https://www.sciencefocus.com/the-human-body/how-hellblade-senuas-sacrifice-deals-with-psychosis/. Accessed 17 Nov 2022
9. Jenkins, H., Squire, K., Tan, P.: You can't bring that game to school! In: Laurel, B. (ed.) Design Research: Methods and Perspectives, p. 334. MIT Press, Cambridge, MA (2003)
10. Abbott, D.: Modding tabletop games for education. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (2019)
11. Csikszentmihalyi, M.: Finding Flow: The Psychology of Engagement with Everyday Life. Basic Books (1998)
12. Hunicke, R., Leblanc, M., Zubek, R.: MDA: a formal approach to game design and game research. In: Workshop on Challenges in Game AI, pp. 1–4 (2004)
13. Vargas, J.A., García-Mundo, L., Genero, M., Piattini, M.: A systematic mapping study on Serious Game quality. ACM Int. Conf. Proc. Ser. (2014)