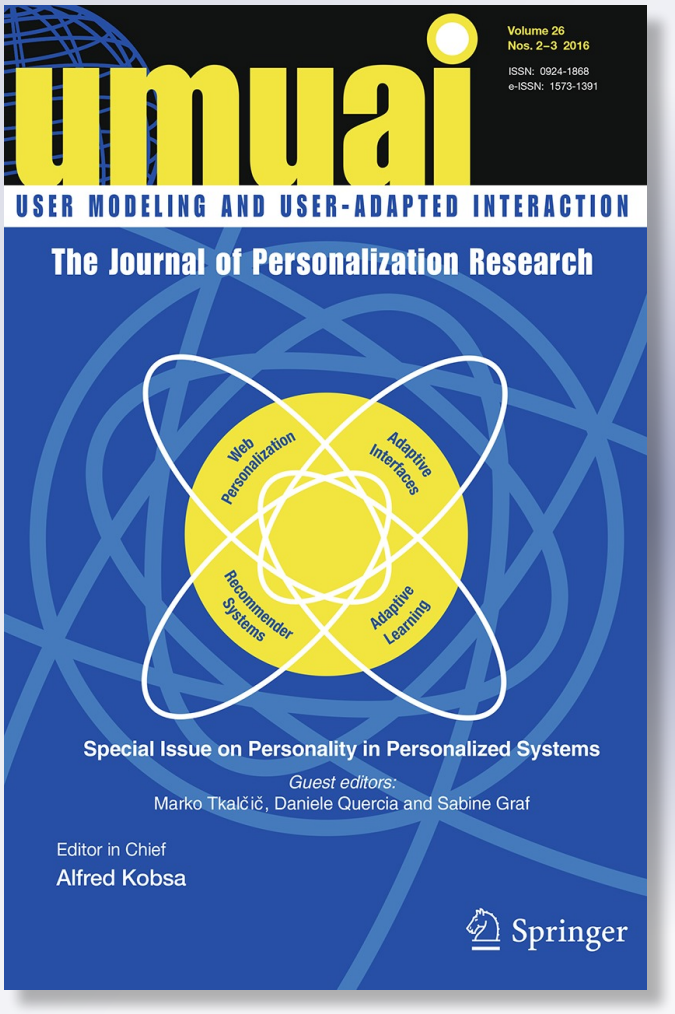
Behavlets: a method for practical player modelling using psychology-based player traits and domain specific features

Benjamin Cowley & Darryl Charles

User Modeling and User-Adapted Interaction
The Journal of Personalization Research

ISSN 0924-1868
Volume 26
Combined 2-3

User Model User-Adap Inter (2016)
26:257-306
DOI 10.1007/s11257-016-9170-1



Springer

User Model User-Adap Inter (2016) 26:257–306
DOI 10.1007/s11257-016-9170-1

Behavlets: a method for practical player modelling using psychology-based player traits and domain specific features

Benjamin Cowley[1,2] · **Darryl Charles**[3]

Received: 3 April 2015 / Accepted in revised form: 8 December 2015 /
Published online: 8 February 2016

Abstract As player demographics broaden it has become important to understand variation in player types. Improved player models can help game designers create games that accommodate a range of playing styles, and may also facilitate the design of systems that detect the currently-expressed player type and adapt dynamically in real-time. Existing approaches can model players, but most focus on tracking and classifying behaviour based on simple functional metrics such as deaths, specific choices, player avatar attributes, and completion times. We describe a novel approach which seeks to leverage expert domain knowledge using a theoretical framework linking behaviour and game design patterns. The aim is to derive features of play from sequences of actions which are intrinsically informative about behaviour—which, because they are directly interpretable with respect to psychological theory of behaviour, we name 'Behavlets'. We present the theoretical underpinning of this approach from research areas including psychology, temperament theory, player modelling, and game composition. The Behavlet creation process is described in detail; illustrated using a clone of the well-known game Pac-Man, with data gathered from 100 participants. A workshop-based evaluation study is also presented, where nine game design expert participants were briefed on the Behavlet concepts and requisite models,

Electronic supplementary material The online version of this article (doi:10.1007/s11257-016-9170-1) contains supplementary material, which is available to authorized users.

✉ Benjamin Cowley
ben.cowley@helsinki.fi

1 Brain Work Research Centre, Finnish Institute of Occupational Health, 00290 Helsinki, Finland

2 Cognitive Brain Research Unit, Cognitive Science, Institute of Behavioural Sciences, University of Helsinki, P.O. Box-9, 00014 Helsinki, Finland

3 School of Computer and Information Engineering, University of Ulster, Coleraine BT52 1SA, Northern Ireland, UK

and then attempted to apply the method to games of the well-known first/third-person shooter genres, exemplified by 'Gears of War', (Microsoft). The participants found 139 Behavlet concepts mapping from behavioural preferences of the temperament types, to design patterns of the shooter genre games. We conclude that the Behavlet approach has significant promise, is complementary to existing methods and can improve theoretical validity of player models.

Keywords Player modelling · Machine learning · Temperament theory · Psychology · Game design patterns · Behavlet

1 Introduction

Understanding and modelling the differences between players can help substantially in the design of games. During the design process, such insight can help to create games that appeal specifically to the target demographic (Herbrich et al. 2007). If dynamic player behaviour is modelled effectively, then it also offers an opportunity to personalise and adapt gameplay in real-time, to enhance the user experience (Chanel et al. 2008).

Game designers have long understood that differences between players should be accounted for in a game if it is to have broad appeal. Providing a choice of difficulty settings for gameplay is the most common approach to account for player variation, from early videogames games such as Tempest (Atari Inc, 1981) through to modern classics such as the Halo series (Bungie, 2001–2010). Game designers may also decide to create a game with one level of difficulty and progressively increase this difficulty throughout the game (e.g. Dark Souls 2, From Software, 2014), recognising the part of the player as a learner and the role of the game as a teacher. It is also not uncommon for a modern game to incorporate Dynamic Difficulty Adjustment (DDA) (Hunicke and Chapman 2004; Hunicke 2005) as in the "rubber band" AI game balancing method in the Mario Kart series (Nintendo, 1992–2014), or the adaptive difficulty of computer controlled opponents in first person shooters such as Max Payne (3DRealms, 2001) and Prey (3DRealms, 2006). Difficulty is a key factor to tune for an optimal player experience, and DDA has resulted in some interesting approaches (Chanel et al. 2008; Missura and Gärtner 2009; Zook and Riedl 2012). However, there are many other factors which impact player engagement. Both player-selected difficulty settings and DDA typically account for variation in capability but not in player type. Features for DDA are not designed to enable a deduction about player psychology but rather to tune the 'game-challenge' utility function.

Player behaviour can be modelled either before or after game release. During development player data can be obtained through play testing, observation, questionnaires, psychometric measurements, and interviews. After commercial release, game analytic tools are employed to monitor game performance, player behaviour, and to learn about the effectiveness of level design (e.g. where players get stuck or stop playing a game). Game and player data metrics are transmitted from game clients to databases on cloud-based servers. Data mining is commonly employed to investigate the game metrics and more recently computational intelligence methods have been utilised to extract

statistical features of play, and cluster or classify players based thereon. Drachen et al. (2009) used the latter approach to build features of play from common in-game metrics (e.g. causes and locations of deaths; completion times), to identify four player types and provide them with personas (Veterans, Solvers, Pacifists and Runners). While this approach is a straightforward way to obtain a player profile, the method gives no guarantee to converge to interpretable clusters. In another approach, Holmgard et al. (2014) designed agents they call "procedural personas", which can then be compared to human play and evolved to emulate it. In both these works, the outcomes do not necessary link well to established theories of personality or temperament.

The motivation for the work we present here is to make it easier to understand *patterns* of player behaviour in computer games or game-like activities, e.g. game-based learning. Player behaviour can be complex, varying and difficult to interpret (Salen and Zimmerman 2004). To reason about this complex player behaviour, it must be possible to make a multi-dimensional mapping from game metrics to theories of behaviour, such that, having a game metric data set one can computationally derive the class of behaviour the player is exhibiting. For example, by contrasting variables in the game which measure exposure-to-risk with variables measuring skill, one may classify players as either cautious or risk-takers. The problem lies in extending the mapping to models non-trivially related to human psychology and thus make valid interpretations of observed behaviour.

In this paper, we describe a method for deriving more informative features of gameplay. A domain expert using the method, such as a game designer, can apply his expert knowledge to create player models *supported by* established theories of player psychology and game structure. Our methodology draws on the temperament theory of human information processing to help the domain expert derive game-specific *features of play* which are more than just gameplay metrics. Rather, such features are a directly interpretable representation of player behaviour, which describe facets of behaviour that expose a personal attitude to playing a game; thus we coin the term 'Behavlets' to describe them.

We define the term *Behavlet* as: a non-trivial informative feature describing observable behaviour in a game, which is related to the persona of the player based on established psychological theory. Behavlets may be thought of as features that help define what motivates a player to make gameplay choices. A simple example of a Behavlet in the classic game Pac-Man (*Namco*, 1980) would be a feature that tracks the player's tendency to attack the opponent Ghosts, which helps to illustrate the psychological disposition toward maximising Power versus Security. We give the full derivation of such a Behavlet in Sect. 4.

Our approach is to introduce two layers of conceptual modelling which aid a domain expert to translate established psychology of behaviour (temperament theory) into Behavlet extraction in the context of a given game. The first layer directly relates game-based temperaments to play traits, as inspired by Bateman et al. (2011). Then, the second layer seeks to assign a set of generic actions to each play trait, inspired by work on game-design patterns (Björk and Holopainen 2005). As a third stage, we suggest an approach to help the domain expert to encode observations of the trait-related actions as Behavlets.

In summary, we propose the Behavlet concept and method to create psychologically-informative features of gameplay. A Behavlet may be constructed on the basis of game design patterns and grammar, knowledge of alternative play styles in a specific domain, and an understanding of variations in player type. This process involves identifying player behaviours over sequences of game state variables from game session log data. Such behaviours can be defined as macro level (i.e. more than one state change) game play actions within repeating situations. In other words, they are patterns of player action that could be observed to be consistent across repetitive game situations by a hypothetical third party observer, where there are three main sources of influence which shape these reoccurring patterns: (a) the human repertoire for engaging with tasks; (b) the kinds of experience which game designers attempt to induce; (c) the actions which are actually possible within a game.

The Behavlet method draws together three lines of work, released in several of the first author's existing publications, so to illustrate the novelty of this paper it is appropriate to describe these foundations. In the first line, Cowley (2009, pp. 77–90) describes how we devised a naïve[1] concept of behaviour traits and the 'constraint harness' approach (defined below) to derive features descriptive of player type. These concepts were applied in Cowley et al. (2013) to show how analysis of Decision Tree player models could benefit from richer features. In the second line, the core Behavlet ideas, inspired by the architectural patterns of Alexander (1990), were developed in the context of serious games development for behaviour change (Cowley et al. 2011). Behavlets in that work described patterns of real-world energy-use behaviour that could be leveraged by simulation game designers, elaborated further in Cowley (2014). Finally, the third line arose from Cowley et al. (2014), in which game design patterns were related to machine-learned clusters of game events. Cowley et al. (2014) showed how disparate methods could be integrated, but itself lacked a means to describe the player's actions in psychological terms, showing the need for the current work. The main contribution of this paper is to define comprehensively the Behavlet process for player modelling. Although other work is referenced at each stage, the working procedures at each stage, and progression between stages, are all novel.

The following Sect. 2 describes the background and logic of our approach, focusing on research into the three 'sources of influence' just described. In Sect. 3 we describe methods for defining Behavlets, working from theoretical models. Sections 4 and 5 present results to illustrate and evaluate the process. In Sect. 4 we derive Behavlets from the game Pac-Man, using game-play data gathered from over 100 players. In Sect. 5 we describe an evaluation experiment which demonstrates the utility of the approach. To close the paper, we discuss our conclusions and some potential future work in Sect. 6.

2 Background

In this section we outline the core literature from the two main supporting areas of research supporting our method: Game Decomposition and Player Psychology.

[1] Defining naïve as being without reference to the theories of psychology used in this paper.

For more general reading on machine learning in games and player modelling, see e.g. Bakkes et al. (2012) or Galway et al. (2009). In Sect. 2.1 we examine several key approaches for describing the composition of games. In Sect. 2.2 we outline background research on player psychology. Section 2.3 covers some other methods which also use a combined approach.

2.1 Game decomposition

To support the definition of Behavlets it is necessary to first decompose a game into its constituent parts. This is a non-trivial task due to the difficulty of defining games in the general case; as Wittgenstein (1953, p. 27) said on the commonality of features within games "you will not see something that is common to all, but similarities, relationships, and a whole series of them at that". We can examine a game from many perspectives: viewed as a formal system built of entities and rules, a game is mechanistic; viewed as an interactive system exhibiting emergent properties, a game is dynamic; viewed as an emotional experience, a game is aesthetic. These mechanics, dynamics, and aesthetics form the three perspectives in LeBlanc's MDA (Mechanics Dynamics Aesthetics) model (Hunicke et al. 2004). In the seminal work of Salen and Zimmerman (2004), seventeen separate perspectives are used to examine games as systems ('Rules schema'), experiences ('Play schema'), and cultural artefacts ('Culture schema'). Salen and Zimmerman also contribute the concept of Constituative (*sic*), Operative, and Implicit rule sets. There is some correspondence in the two frameworks:

- Mechanics: perspectives of Game Theory, Information theory, Systems of Information; also Constituative rules
- Dynamics: perspectives of Cybernetic Systems, Emergent Systems, Uncertainty, Conflict; also Operative rules
- Aesthetics: perspectives of Experience, Pleasure, Meaning, Narrative, Simulation, Social Play

Several authors have considered gameplay patterns from a game design standpoint. For example, Koster (2005) focuses on the inherent fascination that people have for patterns as a motivation to play, outlines how this relates to our desire to learn, and discusses the relationship of learning to the experience of fun. The basic concept is that a game contains patterns of activity that are initially unfamiliar to a player, but progress through the game corresponds to an increased understanding of these patterns, and a concurrent increase in skill. Ideas on game play patterns can be related to information theory and in particular entropy and uncertainty (Salen and Zimmerman 2004). For example, Costikyan (2013) postulates that uncertainty is a core ingredient of games; by this postulate, a player must be unsure of the outcome of a game to maintain interest. LeBlanc (2006) has also discussed the role of uncertainty in the context of Formal Abstract Design Tools (Church 2006) to enhance dramatic tension within games. Patterns of play are also related to game design patterns. Game design patterns are readily identifiable objects of play which form the common core of many different games. The pattern consumption that Koster (2005) discusses relates to the unique composition of game design patterns (and other novel game mechanics) for

each individual game, which generates a unique distribution of information across the space of play and therefore represents a unique experience of learning as players sample the possibility space and estimate the distributions.

Several attempts have been made to develop a more formal approach to game analysis, including early proposals from game designers Church (Church 2006) and LeBlanc (Hunicke et al. 2004). They suggest that more effective analysis of games requires a descriptive grammar of play to underpin a common practical game design language. One approach is to break a game down into its most basic parts, e.g. atoms (Cousins 2005) or to build a practical collection of game design patterns, e.g. the 400 Project (Rouse III 2015). Bethke (2003) used the Unified Modelling Language (UML) to define game elements, giving an example for Pac-Man, among others.

In Chapter 2 of Brathwaite and Schreiber (2009), the authors lay out the elements of a game, including a neat definition of game space as the embodiment of the game, game state as all game variables, and game view as what a player can access at a given moment. Järvinen produced a comprehensive "theory about the parts that games are made of" (Järvinen 2009, p. 45). He defines some useful concepts, such as components "objects that the player is able to manipulate and possess in the course of the game" (Järvinen 2009, p. 63), broken down among components possessed and controlled by oneself, by others, or by the game system. This object-oriented view relates well to category theory (Walters 1991), a method of formal modelling often used for proving computational systems and employed to define games by Grünvogel (2005).

Game designers have consistently used game design patterns over the years, either intentionally or intuitively, and so this is a natural method to use. The game ontology project (GOP) (Zagal et al. 2005) is a simple structure of four categories and one hierarchical level that captures the important structural elements of games and relationships between them. Schell (2008) provides an elegant analytic framework in his 'Book of Lenses'. Björk and Holopainen (2005) have completed some of the most comprehensive research on a complete framework for describing games in terms of game design patterns. Björk and Holopainen describe a game design pattern as "*semi-formalised interdependent descriptions of commonly reoccurring parts of the design of a game that concern gameplay*" (Bjork and Holopainen 2006).

Björk and Holopainen take a two stage approach in their method. They first describe a component framework where invariant aspects of gameplay can be mapped to games. Their framework has the following top-level game components: *Holistic*, *Boundary*, *Temporal*, and *Structural*. Holistic components help define what is unique about a game in comparison to other activities. The boundary component defines the mode and purpose of a game within constraints. Goals are important to many games and are often broken into sub-goals to structure the learning process and challenge for a player, and often to structure a narrative. Temporal components of games define the activity of playing a game, where a player performs a series of actions in the pursuit of goals. Structural components are the most tangible of the four categories, comprising aspects such as game elements that form the basis of the gameplay (e.g. weapons, armour).

The second part of Björk and Holopainen's approach is to define commonly reoccurring design patterns which can fit into the component framework. Components provide an abstraction of a game, but game design patterns describe how specific com-

ponents interact to provide gameplay. The patterns are established using a designer-like approach by collecting and describing different events and components from the game then reflecting on how each one relates to the game playing experience. The result is an ever-growing list of reoccurring patterns in games, which allow us to describe the design of games and gameplay in a comprehensive manner.

In the published collection, the Björk and Holopainen described over 200 patterns found repeatedly across different games. An example is the *Aim and Shoot* pattern, very common in many game types, not only first person shooters. This pattern involves dexterity-based action where one needs to pinpoint a target from a simulated space in real time and then initiate shooting (Björk and Holopainen 2005, pp. 150–153). A different example which demonstrates how design patterns can capture non-sequential elements is the pattern *Perfect Information*. Games utilizing this pattern never hide or keep secret any elements of the game from the player, nor depend on random input, for example Chess or Go (Björk and Holopainen 2005, pp. 128–130). The patterns can be grouped to reflect patterns of similar qualities and scope. Björk and Holopainen cover patterns in eleven major groups, including *resource management*, *social interaction*, *game session*, and *replay value*. The game design template used by Björk and Holopainen contains seven categories which cover descriptions of the pattern, how the pattern may be used, consequences or limitations, and relationships to other patterns (e.g. instantiating, modulating, or conflicting). Lankoski and Björk (2011) have also shown how design patterns can be theoretically derived, extending the value of the system beyond the limits imposed by the previous requirement of manual analysis.

We use Björk and Holopainen's game design patterns as the most practically applicable, comprehensive, and community-supported system. This work is perhaps the most rigourous and comprehesive detailed analysis of games in the literature; it has a research basis and is not simply a list of lists. Design patterns allow us to focus Behavlet analysis on what the player actually does in the game, and they provide a common vocabulary and well-formed structure, which supports the use of modelling tools.

2.2 Player psychology

Beyond the decomposition of a specific game, we must still account for the influence of the player's personality on gameplay.

Koster (2005) proposed that learning is fundamental to a game experience, supported by evidence from educational and comparative psychology (Gee 2003; Groos 1898); this theory indicates that models for describing ordinary differences in learning ability would also serve to describe differences in game playing experience. For example, individuals who are particularly good at analytical maths might excel when game mechanics call for abstract reasoning skills. In one such application, (Acuña et al. 2010) attempted to model the behaviour of expert players of a game based on the Euclidean Travelling Salesman Problem, in order to show that such modelling could find novel solutions to NP-hard problems. Learning style-based models are a potential avenue for future complementary research in player modelling.

Salen and Zimmerman's (2004) implicit rules convey an additional aspect of games related to the Magic Circle concept (Huizinga 1949); that is, the core experience of

games represent an experiential space apart from normal life, which may be thought of as a kind of informal social contract where the normal rules of behaviour do not apply in the same way. This has additional implications, because the standard influences on an individual that arise from the interaction of personality and environment are weakened; within the novel space of a game, players may assume a *play personality* quite different from their own. For example, a player may be more aggressive than they would be in real life. However, players may still be subject to the influence of their basic underlying personal temperament type; it is generally difficult for people to adopt a persona that is fundamentally unaffected by their core personality type—it requires some ability to act. The Behavlet approach is more about capturing styles of dynamic interaction and decision making, than about the suspended disbelief of playing as an avatar. Thus although several player typologies have been proposed, beginning with Bartle's types for online multiplayer games (Bartle 1996), we base our approach on temperament theory.

Temperament theory describes modes of operation of human personality, how we act, react and interact (Berens 2006). Two particular temperament theories (Berens 2006; Keirsey and Bates 1984) are related to the popular Myers-Briggs Type Indicator (MBTI) (Ludford and Terveen 2003) via a similar theoretical foundation, which in general proposes four categories that describe interrelated needs, values, talents and behaviours. For example, Keirsey and Bates (1984) propose an extension of Plato's classic types, Artisan, Guardian, Idealist, and Rational, with two categories and roles per type such that the whole model has 16 subtypes: correlating with the 16 types of Myers-Briggs. However, it is not clear that this level of refinement of personality is applicable to a game playing context.

Berens (2006) proposes four archetypes, which (most importantly here) are related to four skill sets: *Logistical*, *Tactical*, *Strategic*, and *Diplomatic*. Although the theory is not used as widely as trait models such as the 'Big 5' (McCrae and John 1992), there is reason to believe that the two approaches are not strictly incompatible (McCrae and Costa 1989), and types are a more useful approximation than traits when used in this context as a domain-shaping step. Temperament theory is a long established field of research, and as an approach that seeks to understand core motivation it provides a solid theoretical basis and useful skill-preference model in a game context. Temperament theory has been used before in human–computer interaction systems' research (Gómez-gauchía et al. 2006), and has served as an influence in a player typology called 'Brainhex' (Bateman et al. 2011). Each skill set is associated with preferred behaviour: this is the key information on which to build our Behavlet method, expanded further in Sect. 3.2. Table 1 is adapted from the work of Stewart (2011); it illustrates how richly the temperament theory can be mapped to existing game player models, thus supporting the general validity of our approach.

2.3 Combined approaches

Game decomposition and player psychology models must be combined for maximum efficacy. This requires going beyond consideration of separate perspectives, e.g. MDA, to the definition of a framework that links perspectives with a scientifically plausible theory. For example, such a framework should account not only for the learning

Table 1 Mapping of four temperament types (Keirsey and Bates 1984) to a subset of four-quadrant game player type models from the literature (Bartle 1996; Caillois and Barash 1961; Hunicke et al. 2004; Lazzaro 2008); as well as Stewart's own conception of associated motivation, problem-solving style, and overall goals (adapted from Stewart 2011)

Keirsey	Bartle	Caillois	Lazzaro	MDA+	Motivation	Problem-solving style	Overall goal
Artisan (tactical)	Killer	Ilinx	Serious fun	[kinetics]	Power (manipulative sensation)	Performance	Do
Guardian (logistical)	Achiever	Agôn	Hard fun ('fiero')	Mechanics	Security (competitive accumulation)	Persistence	Have
Rational (strategic)	Explorer	Mimesis	Easy fun	Dynamics	Knowledge (logical rule-discovery)	Perception	Know
Idealist (diplomatic)	Socialiser	Alea	People fun	Aesthetics	Identity (emotional relationships)	Persuasion	Become

that occurs throughout a game, but also the emotions of players and their impact on play.

The User-System-Experience (USE) model was an early attempt at such a framework proposed by the authors (Cowley et al. 2006, 2008). In this framework the most novel aspect was the formulation of the pleasurable and autotelic nature of games in terms of Flow theory and the neurobiology of information processing and learning. However the specification of games themselves was lacking in detail. Another method, similar to what we propose here, has been demonstrated for the domain of educational games (Bedek et al. 2011; Cowley et al. 2012); here players were modelled in terms of the competences they show in the pedagogical domain. Competence model validity is improved by theory-driven expert design of features, because the psychological underpinnings of real-world competences are defined in the literature. This method works hierarchically from a description of a competence such as communication; through sub-competences, such as verbal and non-verbal communication; to a set of behavioural indicators based on empirical findings, e.g. non-verbal communication is indicated by listening, body language, proxemics (personal space); to contextual performance indicators represented by a formula defined over in-game variables, e.g. for proxemics there is position coordinates, viewport, etc. Such complexity shows that domain expertise is often needed in creating serious games (Marchiori et al. 2012), but less has been done to systematise that knowledge for player modelling, although techniques from intelligent tutoring have long served as inspiration for profiling in entertainment games (Beal et al. 2002). Herbert and colleagues took a similar approach to build a model of a gamified learner using gamification types based on combined psychology and gamer types, also tracking behaviour within a learning virtual world to investigate and refine the model (Herbert et al. 2014).

Järvinen (2009, pp. 99–247) proposed a model of player experience that builds on his game decomposition theory. Central to this model are two concepts: that game experiences are composed of sequences of emotions; and that game elements embody conditions that elicit emotions. It is also important that emotions are part of the cognitive game; the player is seen as predictive of her own and other players' emotions, which forms a reciprocity between emotion elicitation and active play. Gmytrasiewicz and Lisetti (2000) also attempted a formalism of 'synthetic' emotions using Decision Theory, to be used for player modelling or for communication of AI agent states to the player. However, as Grünvogel (2005) says, such formal system-based definitions of games can be cumbersome because they attempt to encapsulate all aspects of all possible games in one system, becoming either unwieldy or insufficiently descriptive. This is one reason why we propose a multi-level approach. In previous iterations of this work (Cowley 2009, p. 82) we used a three-step process for capturing game mechanics: natural language descriptions, pseudo-coding and game engine coding. To improve transparency and allow replication of the process, in this paper we contribute a much more finely detailed, iterative 'process definition' for how to build Behavlets.

3 Methods—Behavlet framework

In this section we outline a framework for identifying, embedding and tracking Behavlets. Game components, design patterns and a fundamental model of psychology form a foundation for this framework; instantiated via notation for modelling that supports the coding of features in the game engine. The theoretical foundations relate to each other as shown schematically in Fig. 1, which is a flow chart for Behavlet extraction.

Top (theoretical): the method draws on comprehensive and detailed design patterns, paired to a personality/temperament model with mapped play/game characteristics.

Upper middle (analytical): the method considers the game in question as 'ground truth', and utilises structures from stage 1 to describe core play traits and game struc-

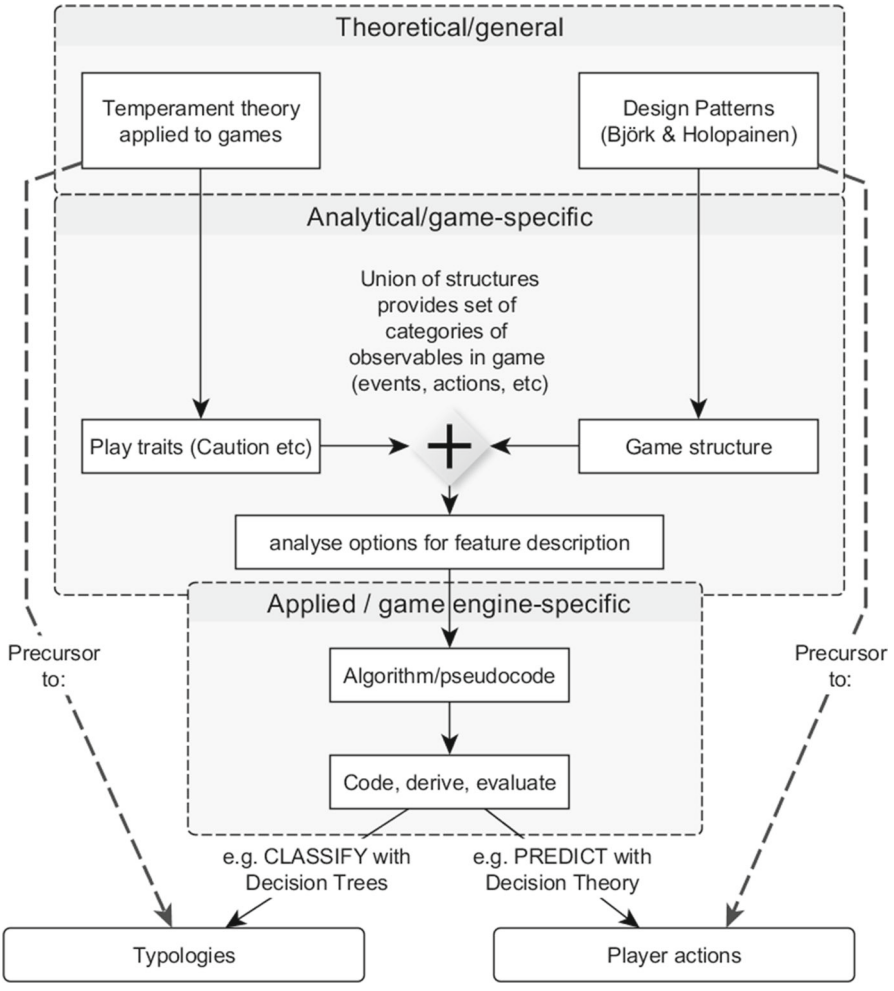


Fig. 1 Theoretical relational framework for Behavlet extraction

ture. Lower middle (applied): these gameplay features, which we term Behavlets, are codified as pseudo code, and integrated into or alongside the game for tracking.

Bottom (classify/predict): in our implementations of Behavlets, the adaptive phase involves predicting player actions from player models comprised of Behavlet features and classifying or reclassifying type based on accuracy of prediction.

The Behavlet approach is a way to model players based on variation of their dynamic gameplay behaviour. Behavlets comprise track-able player actions or sequences of actions, which provide insight into a deliberate decision making process. Behavlet profiles vary between players based on personality and may be dependent on other factors such as game type (hence the importance of an adaptive model). In this way the Behavlet model somewhat resembles a Hidden Markov Chain model. For example, in a computer game a player can usually move an avatar or some pieces. The *way* they move very often has qualifiers, such as fast/slow or cautious/aggressive movement, and these qualifiers can have a basis in player temperament which is not directly observable. To extract and characterise Behavlets, i.e. features codifying these behaviours, we start with the structural foundations of temperament theory and design patterns. These structures are refined to the specific case of the game, and the two structures are combined to give a lens through which the game is analysed, searching for feature descriptions. Identified features are given a working description in natural language. The working description is translated to a coded form; data is gathered, resulting in a Behavlet feature set; this set can be subjected to feature selection methods.

The entire process has four stages, containing respectively three, three, one and one modelling steps. The following description and flow chart, Fig. 2, outlines an iterative process for a game designer/developer (GD) and associated team:

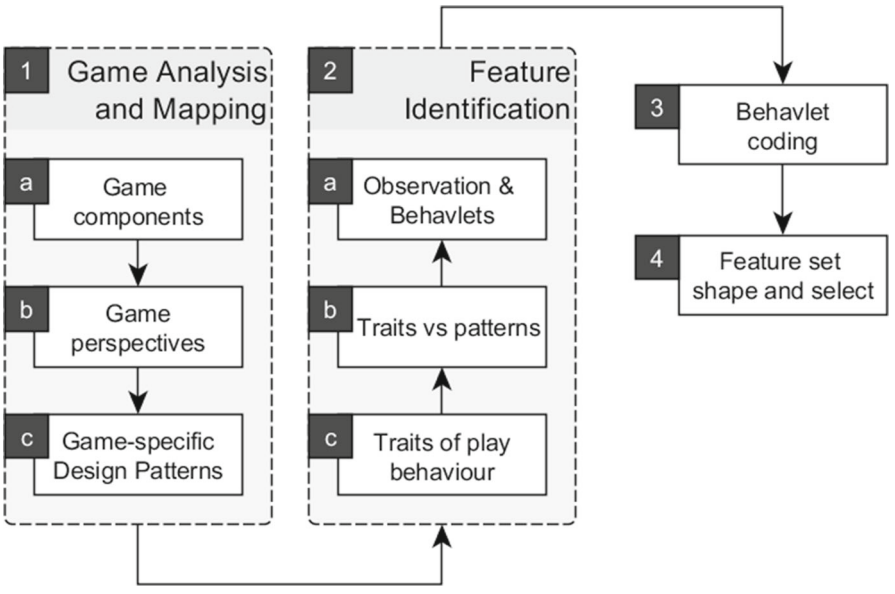


Fig. 2 The four stages of theory-driven Behavlet extraction

Table 2 Components from the game structure model of Björk and Holopainen (2005); with those sub-components selected that satisfy the constraint *provides agency*

Component	All sub-components	'Provides agency' sub-components
Holistic	*Game Instance* → Setup/Set-down, *Game Session* → Setup/Set-down, *Play Session* → Setup/Set-down, *Extra-game* → Activities	*Play Session*
Boundary	*Rules*, *Modes of Play*, *Goals* → Sub-goals	*Goals, Play Mode, Rules*
Temporal	*Actions*, *Events*, *Closures* → Sub-closures, *Conditions* → End, *Functions* → Evaluation	*Actions, Events, Closures, Conditions, Functions*
Structural	*Interface*, *Game Elements*, *Players*, *Game Facilitator*, *Game Time*	*Interface*, *Game Elements*, *Players*, *Game Time*

(1) *Gameplay analysis and mapping*

(a) Game structure and context (*leads to*) →
(b) Game mechanics and dynamics →
(c) Game design patterns.

In this stage, GD should identify which game components deliver agency to the player, and thus are central to describing player behaviour. Further, GD must differentiate between game mechanics, the building blocks of game-play described by Constituative rules, and the dynamical operations of player–game interaction. Then design patterns specific to the game can be identified.

(2) *Feature/behavlet identification*

(a) Traits of play behaviour →
(b) Traits versus patterns →
(c) Observation and Behavlets.

In this stage, GD specifies how behaviour would express itself. GD uses a list of descriptive terms for behaviour (given in Tables 2 and 3), to characterise how design patterns would turn into extended sequences of action selection. GD then observes the play of the game to develop Behavlet concepts.

(3) *Behavlet coding*

For the informal Behavlet set from step 2, GD then defines pseudo-code and game engine encodings.

(4) *Feature set shape and select*

GD performs feature selection to test individual Behavlet contributions and cull uninformative ones.

The process requires several theoretical contributions which follow in the next three sections. We define:

(1) A simple method of game specification based on existing theories, centred on design patterns.

Table 3 Description of player archetypes and their preferred play styles

Player Archetype	Drawn to…	Behaves with…	Tolerant of…
Logistical	Optimization, planning, trading	Caution, meticulousness	Repetition, rules, procedures
Tactical	Improvisation, thinking on the spot, operation, controlling single characters	Impulsiveness, competence	Risk, speed, variation
Strategic	Solving, hypothesizing, controlling multiple units, thinking ahead	Logic, perfectionism	Complexity
Diplomatic	Harmonizing, imagining, co-operation	Empathy, morality	Impressionism

Adapted with permission from Bateman et al. (2011)

(2) Behavioural traits of play based on temperament theory.
(3) An approach for linking psychology to specification, and defining game play actions.
(4) *We make no contribution in the area of feature shaping/selection*: many methods exist in the literature.

3.1 Gameplay analysis and mapping

In the first instance we need to understand a game's design in more holistic manner; considering the game's broader play, gaming, and cultural context. We propose an emphasis which draws upon three existing methods for modelling games and game-play. The process is not strictly prescriptive which is why some domain expertise is important.

3.1.1 Game structure and context (descriptive focus)

In the first step, appropriate game components are chosen based on Björk and Holopainen (2005)'s model, shown below; these components help to define a game in terms of its whole context. They must satisfy the constraint relation *provides agency*: i.e. chosen components contain all game elements that enable the player to explore the possibility space of the game; all others are excluded. The set of sub-components chosen in Table 2 are a first approximation—any specific case requires reassessment.

3.1.2 Game mechanics and dynamics (analysis and design focus)

The second step further refines the lens on the game, by adopting in turn each of the core perspectives from Sect. 2.1: Mechanics/Constituative rules, and Dynamics/Operative rules. The goal is to clarify the *scope* of game components: to differentiate between the rules of the game and the dynamical operations of game-play. Mechanics can

be expressed in terms of *atoms* of play (Cousins 2005); while Dynamics arise from an emergent process, which implies that they must be expressed only over several atoms: i.e. the *scope* is greater. Therefore, the scope of player and game actions must be defined, to distinguish those that can characterise player behaviours from more simple ones. The procedure for this is to identify, for any given action-oriented design pattern, the simplest possible atomic action sequence for the player, forming a loop based on Perception → Analysis → Decision → Execution (PADE), with associated feedback coming from the game. This identified loop then enables discovery of more complex interactions, i.e. atoms versus molecules of play, which better serve to express temperament-driven behaviours.

3.1.3 Game design patterns (implementation focus)

Design patterns are specific, tried and tested, blueprints for implementation of facets of gameplay. We utilise game-specific design patterns following the methodology of Björk and Holopainen (2005), although strictly limited to the components and perspectives identified in steps one and two. Eleven pattern groups are provided in their method, from which we select the most relevant to list below (because there are over two hundred patterns, we do not list them individually here). These should be followed as listed, in order of importance to the expression of player behaviour, identifying patterns group by group.

1. Actions and Events Patterns
2. Game Design Patterns for Social Interaction
3. Game Design Patterns for Game Mastery and Balancing
4. Game Design Patterns for Game Elements
5. Game Design Patterns for Resource and Resource Management
6. Game Design Patterns for Information, Communication and Presentation
7. Game Design Patterns for Narrative Structures, Predictability, and Immersion Patterns
8. Game Design Patterns for Goals
9. Game Design Patterns for Goal Structures
10. Game Design Patterns for Game Session
11. Game Design Patterns for Meta Games, Replayability, and Learning Curves

A game can potentially be analysed and broken down using the above three stages in an iterative cycle, especially for a complex multimodal game: as each step is completed, insights into the game will grow, which will enable further specification at earlier steps. The outcome of analysis is gaining an understanding of a game, its design, play context and gameplay, so as to be able to be able to express the game in a detailed and practical manner.

3.2 Behaviour identification

We propose a further three-stage process to link the predispositions for action described by Temperament theory to the structural decomposition of gameplay elements as

described above, e.g. patterns of game design described by Bjork and Holopainen (2006). These three stages facilitate the identification of Behavlet-type observable patterns of action, which are (as explained) like psychologically-grounded features of gameplay,

3.2.1 Traits of play behaviour

The relationship between the four archetypal skill sets of Temperament theory and the playing of games is shown below in Table 3, adapted from Bateman et al. (2011). The behaviour of these four player archetypes is marked by the *abstract* nouns given in the third column, which indicate how the play behaviour of each player type would be modified by that type's preferences/tendencies. Given that type systems assume that every individual embodies each of its types to some degree, these behaviours are not to be taken as mutually exclusive. As mentioned above (Sect. 2.2), such behaviour is the key information on which to build our feature extraction method, going beyond the common verb-based description of gameplay; this behaviour serves to help target the search for descriptive patterns of play that can be coded as features.

Our method derives a list of high-level behavioural *traits* (from the 'Behaves with…' column), which can be expressed, either positively or negatively and to varying degrees, in most games and by most players. Play traits describe ways in which players explore the possibility space, reducing the uncertainty towards the game's outcome. In this sense they are related to player types, which describe the way games can be enjoyed. Importantly, they are instantiated from a sequence of action selection choices, as these embody the mechanics of how the player explores the space/reduces uncertainty.

We are proposing a practical method to extract features, therefore we relate the behaviours more directly to play by providing synonymous terms which may be closer to the context of game play, easier to define in terms of the actions and reactions of play, and thus clearer for the practitioner/designer. All synonyms are given from Collins thesaurus. In Table 4, the 'Applied Form' denotes the term we prefer as best suited to the gaming context, and use throughout the test.

Our trait list thus comprises Caution, Thoroughness, Aggression, Decisiveness, Planning, Optimisation, Empathy, Fair Play, Resourcing, Speed and Control Skill. Separating these generic traits from game-specific features divides the space of reasoning about player behaviour into different levels of specificity and different focus areas in each game. This helps to build a method that is useful across a variety of game types.

Most games emphasise one or more of our traits, but few games force the player to adopt a particular style of playing; these traits therefore differ from the concept of a control scheme or modality. If a play style is forced by the game, then it is no use for distinguishing between players since all must play the same way.

Our high-level game-play traits were specified with regard to a concept of single-player, console-type play in contemporary games. Any game quantity not controlled or affected by players, whether or not it is recorded in metrics, should not be assigned to a trait. Since the emphasis is on ways that players express themselves in the game and how they interact with the mechanics of a game, their physical reactions (e.g.

Table 4 Table adaptation of temperament type behavioural styles to a form suited to application

Original	Applied form	Description	Other useful synonyms
Caution	*Caution*	Describes how much the player guards their avatar, lives, or other game-state representation of failure risk	*Watchfulness, alertness*
Meticulousness	*Thoroughness*	Is attempting to do or see everything available, also solving puzzles by exhaustion of combinations	*exact, painstaking*
Impulsiveness	*Aggression*	Describes forward or hasty action with respect to opponents or obstacles	*Impetuous, rash*
Competence	*Decisiveness*	Describes players who don't backtrack or vacillate; whether or not they have planned their play, they are confident in their abilities. This behaviour also relates to control skill, but the distinction is that decisiveness operates on a larger scale	*Proficiency, capability*
Logic	*Planning*	Is achieving higher level goals with a premeditated, linked series of actions	*Reasoning, deduction*
Perfectionism	*Optimisation*	Would often be a meta-behaviour, of replaying the game to gain higher and higher scores	*Purist, stickler*
Empathy	*Empathy*	Is playing well with others, being a good team-mate and engaging whole-heartedly in the role-playing requirement of any game; should not be confused with compassion	*Understanding rapport*
Morality	*Fair play*	In play means observing the rules and refusal to engage in win-at-all-costs behaviour, which damages the experience for the group, or at least those who are not also capable of such behaviour	*Decency, honesty*
–	*Resourcing*	Is the capability of the player to manage the game economy; this 'economy' can be ammunition and health in combat games, items and virtual currency in role playing games, card counting in card-deck games, or any other fungible storage mechanic for player agency	–

Table 4 continued

Original	Applied form	Description	Other useful synonyms
–	*Speed*	Is primarily how fast the player completes discrete segments of game play, be they levels, objectives or even individual moves (as in chess)	–
–	*Control skill*	Means fine, precise and detailed command of control schemes, such as accuracy in a shooter or knowledge of hotkeys in an RTS	–

The last three traits are considered necessary to reflect aspects of play which depend on learning. They do not derive from Temperament theory as they are not general forms of behaviour but rather context-specific forms, defined only in a bounded domain which affords learning, i.e. a game

facial expression, galvanic skin response, other biometrics) were not considered (this issue was addressed in Cowley et al. 2014).

3.2.2 Traits versus patterns

Each of the Temperament theory archetypes also comes with a list of activities they are drawn to, given in column two in Table 3. This list acts as a guideline to relate behaviour traits to the game design patterns of Björk and Holopainen (2005), as exemplified in Table 5. Patterns are polymorphic templates for the design of potential transitions through the game's possibility space, meaning they can be varied from instance to instance—e.g. the pattern 'Limited Resources' can imply quite different gameplay experiences from game to game, as for example the limited health and ammo in 'survival horror' games creates a strategic pressure, compared to the limited time for planning in 'online shooter' games, which creates time pressure. Thus patterns are the abstract expression of the behaviour defined by traits. Due to the large number of patterns, an extensive correlation with behaviour traits can only be done in the bounding context of a given game—but for reference, the example in Table 5 is expanded in Appendix E in Supplementary Material with seven pattern categories per preference, and over 600 associated patterns. This should supply the basis for any mapping of a computer game's patterns to the archetypes.

The play patterns complement the traits in more complex games; they allow us to triangulate the meaning of a Behavlet by relating it to both a behaviour trait and a set of commonly-correlated features, i.e. a design pattern.

3.2.3 Observation and Behavlets

At the end of the third stage, we develop the set of Behavlet concepts, each relating to one or more trait-related play patterns. This has an observational component, working from the game analysis given above and searching for and registering distinct patterns of events. Observation can be done by watching game sessions, examining the game log, visualising game log data, etc. During observation, it will also be likely that a list of simpler 'features' of play are described. These can help when designing/coding Behavlets.

The intent of the player in performing each action sequence needs to be associated with a behaviour trait. The link between player intent and the expression of intent through player action should be as clear as possible, i.e. involve as little ambiguity as possible. Behavlets are thus subject to a condition, which we term the *reasoning condition*, defined as: the requirement that one must be able to reason over (or induce from) the raw game log data in order to derive a behavioural representation.

This condition implies that, without yet making exact logic statements, we need to consider whether it is possible to build a 'constraint harness' for each pattern. We define the term *constraint harness* as: a logic statement defined over the game engine variables which, when triggered, indicates the start of the relevant trait-related play pattern.

If it is clear that the pattern can be captured thus, an important question should be answered before including the Behavlet in the list: can it generate data? Alongside

Table 5 Non-exhaustive list of behaviours that each Temperament theory archetype is drawn to, along with associated design patterns grouped by category (the Pattern Category PC is shown in the right-hand column, where GE means Game Elements and PG means Patterns for Goals)

Archetype	Archetype preferences	Associated game design patterns	PC
Logistical	Optimisation	Extra-Game Information, Ghosts, High Score Lists, Lives, Mule, Pick-Ups, Geometric Rewards for Investments	GE
		Configuration, Exploration	PG
	Planning	Cameras, Extra-Game Information, Ghosts, Goal Points, Obstacles, Pick-Ups, Resource Generators, Traces, Units, Producer-Consumer, Resources	GE
		Capture, Conceal, Configuration, Connection, Delivery, Eliminate, Evade, King of the Hill, Reconnaissance, Rescue, Stealth, Survive, Traverse	PG
	Trading	Cards, Mule	GE
		Gain Ownership	PG
Tactical	Improvisation	Controllers, Enemies, Game World, Role Reversal, Inaccessible Areas	GE
		Evade, Rescue, Survive	PG
	Thinking on the spot	Cameras, Enemies, Game World, Inaccessible Areas	GE
		Collection, Delivery, Eliminate, Evade, Gain Competence, Herd, King of the Hill, Last Man Standing, Race, Rescue, Survive, Traverse	PG
	Operation	Buttons, Controllers, Game World, God's Finger, Levels, Producer-Consumer	GE
		Alignment, Capture, Collection, Conceal, Configuration, Connection, Eliminate, Evade, Gain Information, Guard, Herd, Last Man Standing, Race, Stealth, Traverse	PG
	Controlling single characters	Avatars, Controllers, Enemies, Game World, God's Finger, Obstacles	GE
		Capture, Collection, Conceal, Delivery, Eliminate, Evade, Exploration, Gain Competence, Gain Ownership, Guard, Herd, King of the Hill, Last Man Standing, Race, Reconnaissance, Rescue, Stealth, Survive, Traverse	PG

Table 5 continued

Archetype	Archetype preferences	Associated game design patterns	PC
Strategic	Solving	Clues, Extra-Game Information, Game World, Goal Points, Obstacles, Geometric Rewards for Investments, Resources	GE
		Alignment, Configuration, Connection, Evade, Gain Competence, Gain Information, Rescue, Stealth	PG
	Hypothesizing	Extra-Game Information, Game World	GE
		Alignment, Configuration, Gain Information	PG
	Controlling multiple units	Controllers, Enemies, Game World, God's Finger, Obstacles, Parallel Lives, Units	GE
		Capture, Delivery, Gain Information, Gain Ownership, Guard, Herd, Traverse	PG
	Thinking ahead	Cameras, Enemies, Game World, Goal Points, Role Reversal, Resources	GE
		Alignment, Capture, Collection, Conceal, Configuration, Delivery, Evade, Rescue, Traverse	PG
Diplomatic	Harmonising	Lives, Parallel Lives	GE
		Exploration, Survive	PG
	Imagining	Alternative Reality, Enemies, Extra-Game Information, Game World, Outstanding Features, Tools, Traces, Meta-Games, Inaccessible Areas	GE
		Conceal, Configuration, Exploration, Stealth	PG
	Co-operation	Extra-Game Information, Goal Points, Construction	GE
		Capture, Delivery, Eliminate, Guard, King of the Hill, Rescue, Survive	PG

testing, this question depends on whether the constraint harness is internally consistent and feasible to run. If so, further effort should be given to characterizing the Behavlet. It should be given a short but descriptive name. One must decide if it is atomic or molecular, because play patterns reflective of traits will tend to be more complex than the atomic level.

Once defined, it is useful to consider whether the Behavlet relates to more traits than it was originally conceived for? This is likely to happen and even to generate highly correlated Behavlets, but that is acceptable and indicates good pattern coverage has been achieved.

Finally, for each observed Behavlet, a description of the observed pattern of play and associated trait is written in natural language. The goal here is to guide the algorithm development and coding that follow.

3.3 Behavlet coding

In this section we give guidelines for free coding of the Behavlets. Following the observational step, using a simple encoding scheme, we can generate a working representation of the actions that players can take, and the archetypal forms of behaviour that can shape those actions. Of course, coding the Behavlets depends largely on project-specific issues such as game engine language etc. Our scheme therefore only addresses how to turn the natural language into the appropriate algorithm to represent the Behavlet and capture the observed behaviour.

The basic algorithm, for a given Behavlet β, is (for several simple examples, see Table 11):

- For all states S and variables V defined for the constraint C on β, iterate through all states
  - If a given set of state variables, s of v, match parameter values defined in C
    - Measure β using measurement function f, an operation defined for:
      - States $S2$, variables $V2$, and the β cumulative variable cv.

To state this in plain English, iterating through the states from a raw data log file, the algorithm searches for constraint harness parameters to satisfy. These may take the form of one or many variables at some state, or at temporally distributed states. The constraint harness is a set of logic statements, and the parameters are simply numeric thresholds. The measurement function is likely to be a simple increment count, such that when the constraint is met, the variable for β is increased by one. So β represents a simple count of the number of times the play pattern has been performed. Or more complicated functions can be defined, such as a measurement of another set of state variables, separate to the constraint.

The procedure of algorithm specification involves deconstructing the Behavlet operation to the atomic level of state transitions. Once the state transitions required for the pattern are found, associated variables in a temporal pattern can be defined, and the constraint can be written, along with the measurement function.

Of course it is important to perform quality assurance during this procedure: to estimate the computability of the algorithm, its speed, and potential execution hazards (particularly when the constraint triggers on earlier states than the measurement

function). These issues are simpler to handle when developing algorithms within the game engine, which will already provide the needed functionality for handling raw game log data.

Finally the algorithm description should be re-written using pseudo-code. The goal is to move toward an executable specification of the Behavlet, while still expressing concepts in terms of the player behaviour and not yet the engine code. As Dalbey says "*the vocabulary used in the pseudo-code should be the vocabulary of the problem domain, not of the implementation domain*" (Dalbey 2012). Dalbey also advises that the pseudo-code logic should operate at the level of a single loop or decision, so it can reflect the 'atoms' of gameplay already found and analysed in step 2 (Sect. 3.1.3). The pseudo-code should be simple and clear to read, and because the final coding of the Behavlets will be done in the language of the game engine, selecting a style close to natural language helps to avoid over-specification. Thus we suggest Structured English, a forerunner of the database query language SQL which should be familiar to many game designers (Davis 1983, p. 337).

4 Results—Behavlet analysis of Pac-Man

To summarise Sect. 3, Behavlet development entails: describing the game in terms of player traits and design patterns to give a working understanding of play patterns; then finding and observing play patterns in the logs of played games, associating with traits, and coding each as an algorithm. We now show how the Behavlet extraction method can be applied, following the procedure and theory described above. We use a version of Pac-Man as an example, because Pac-Man satisfies the *reasoning condition* (from Sect. 3.2.3), and offers other advantages as a test bed. For this process we derive an initial set of game-specific features from a test-bed game, as follows:

- We define a Pac-Man-style game, and obtain a data set from 100 players (for details see below).
- Through iterative analysis of game-log data we define the Behavlets and features of play, using the procedure given above (the subsections below follow the same numbering as in Sect. 3):
  1. gameplay analysis and mapping;
  2. feature/Behavlet identification;
  3. Behavlet coding;
  4. feature set shape and select

Participants were recruited within the host institution and by online advertisement; participation was anonymous, voluntary and non-remunerated. Ethical approval for the work was obtained from the Research Governance board of the University of Ulster. After excluding short games (of only one level) we had exactly 100 players (23 females, 77 males, aged 17–41 years, mean 21.4). Players described their previous experience with Pac-Man according to four categories: No prior experience 9 %, Beginner level experience 69 %, Intermediate 14 %, Expert 8 %. Respondents participated using our game software (incorporating a client-server back-end for logging game data in real-time) either remotely or in person at the host university. Respondents were initially presented with an online information screen. This informed them how to use the Pac-

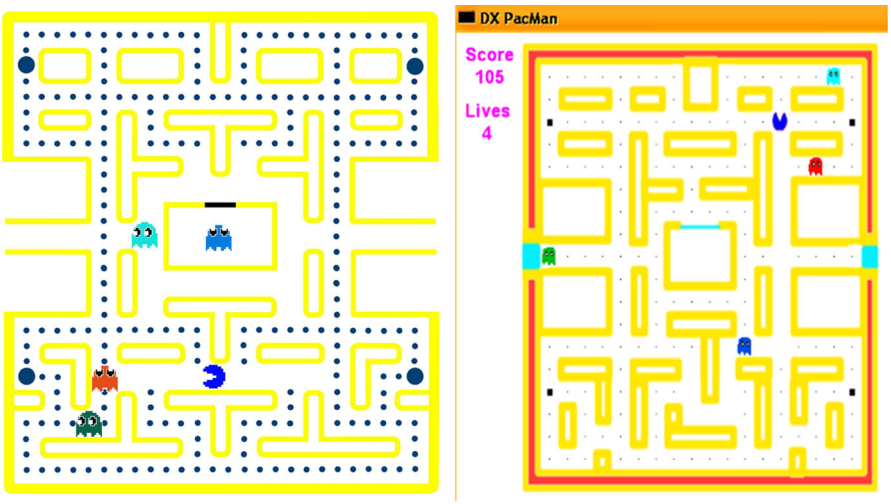


Fig. 3 Screenshots (colour-inverted to minimise ink) of Pac-Man, original version on the *left*, our test-bed version on the *right*. (Color figure online)

Man game. They were advised to play several times to familiarise themselves, but their total number of games was discretionary since only one game was actually used for Behavlet extraction (the game with most levels, from the third repetition on).

4.1 Gameplay analysis and mapping

The first step is to define the game, and understand how it shapes behaviour. Several aspects of the game help ensure that test players should be challenged only by core mechanics: Pac-Man is quite well-known, the control scheme is simple, the game space is completely represented onscreen, and it is of the class of games known as predator/prey, so the more obvious goals/mechanics of play are clear.

Simplicity of elements should not preclude complexity of play, to allow variety in player behaviour which can facilitate expression of temperament. Pac-Man contains both immediate and evolved tactics, which can be used to differentiate players. The 2D grid-like game space allows efficient representation and reasoning. Although agents in the game move continuously, their movement can be quantised in the background, which allows certain turn-based logic such as tree search to be applied. The Pac-Man version we used featured slightly different graphics than the original Namco game, as pictured in Fig. 3. A complete description of that game is given in Cowley (2009), along with the variables recorded in the game log; while Appendix D in Supplementary Material gives an overview of the original Pac-Man in terms of Björk and Holopainen (2005)'s game components. Such high-detail data is important to the Behavlet process, but is highly specific to each game.

4.1.1 Pac-Man structure and context

Björk and Holopainen (2005) provide an example analysis of Pac-Man using their component framework, included in Appendix D in Supplementary Material for refer-

ence. From their complete analysis, in Table 6 we select only those components which meet the '*provides agency*' constraint, as per the method.

4.1.2 Pac-Man mechanics and dynamics

Beginning with the *atom* level of Pac-Man, there is only one simple PADE loop:

- Perception—player scans the map, noting relative position of Ghosts and remaining collectables
- Analysis—the player must assess the threat versus opportunity level associated with up to five courses of action: not moving, or moving in each of up to four directions
- Decision—the player must decide in which direction to move
- Execution—the player pushes the corresponding button

Based on this loop, the Mechanics of the game are: traverse the game map using direction controls to guide Pac-Man, with an imperative to avoid or chase the Ghosts which flips depending on the mode. This remains more or less invariant across the whole game. The Dynamics are more interesting, because they uncover the 'hidden game' which exists only at higher skill levels. Now, although we write another list in the form of a PADE loop atom, each list item is a temporally extensive dynamic of the game which links two or more atoms together. The dynamic is also constantly evolving, so it cannot be simply considered as a longer version of an invariant mechanic.

- Perception—a skilled player notes the spatial relationships inherent in the power play mechanics, i.e. the location of Power Pills which allow Pac-Man to go on the attack
- Analysis—a skilled player will consider the extended consequences of each move as far as possible
- Decision—a skilled player also decides whether and how to adjust his existing plan
- Execution—a skilled player prepares his hand for the next response

4.1.3 Pac-Man design patterns

Design pattern analysis of Pac-Man helps to clarify the structure of the game; it also helps to understand the patterns of activity in observed games. Again, Björk and Holopainen (2005) provide the starting point, having identified several of the Pac-Man patterns:

- the holistic category covers *Lives*, *High Score List*, and *Meta Games*;
- the boundary category provides the goals of *Collection*, *Levels*, and *Role Reversal* (through the Power Pill);
- the temporal category gives *Time Limit*, *Movement*, *Geometric Rewards for Investments* (regarding the number of captured ghosts);
- and the structural category identifies *Enemies*, *Pick-Ups*,*Producer-Consumer*, *Resources*, and *Inaccessible Areas*.

Table 6 Selection of Pac-Man '*provides agency*' sub-components from the Björk and Holopainen (2005) analysis

Component	'Provides Agency' sub-components
Holistic	The play session coincides with the game session in single player mode (two player mode is not supported in our test)
Boundary	Pac-Man has several layers of goals: eating a single Pill to obtain a score (sub-goal), eating all of the Pills to complete a level (goal), and a top level goal to obtain a high score
	Power Pills and fruit bonus points provide optional goals and enhance strategic play. Power Pills change ghost the state from chasing to avoiding and allows Pac-Man decide to pursue and eat them so as to increase a high score or to use this state change to make it easier to complete the level
	If a player decides to hunt ghosts while they are in an evade state then sub-strategies may be adopted such as trying to catch all ghost, or only those in an immediate area
	Modes of play primarily consist of eating Pills while avoiding ghosts and chasing and trying to eat all of the ghosts
Temporal	Closures/conditions/evaluation
	Reaching any of the goals described above may be associated with an achievement closure
	The end closure for eating a Pill is that Pac-Man has moved over it, it is removed from the game, and an evaluation function adds 10 points to a players score
	The goal of eating a ghost has a similar end condition
	Pac-Man colliding with a ghost has two possible immediate outcomes. If the ghost is in evade state the player is awarded points whereas if the ghost is in a chase state the player will lose a life
	Each time 10,000 points are obtained by the player a new life is awarded
	The closure associated with losing all lives results in the game ending
	Actions
	Actions within the game are quite low level. Pac-Man can move up, down, left and right. Movement continues in the same direction until the player changes direction or Pac-Man is stopped by a wall
	Events
	Many events, as every action (particularly due to movement) has a direct effect on a player perception of the game state
	Change of mode of play and closure result in an event
	Events related to how ghosts move (underlying artificial intelligence), and are regenerated

Table 6 continued

Component	'Provides Agency' sub-components
Holistic	The play session coincides with the game session in single player mode (two player mode is not supported in our test).
Structural	Player controls the game by choosing game mode and altering the game state by moving Pac-Man around the maze to satisfy general and personal goals
	Dual chase/evade game state mechanics of the ghosts which alter the play dynamic
	Game elements
	Walls, side openings, ghost 'house'
	Pac-Man avatar
	Four computer controlled ghosts
	Consumables: Pills, Power-Pills, occasional bonus fruit
	Pac-Man lives
	Game levels
	Score to represent achievement and progress
	Interface
	Physical IO devices: Screen, keyboard
	Screen display: display of game elements for a given state (e.g. ghost colours) and mode, information on game state such as score and level
	Inputs are coupled to low level control (via a joystick) of Pac-Man through the maze in four directions until stopped by a wall. Placing coins in a slot and pressing buttons to select game mode

Each pattern can be described in detail (though different patterns vary in complexity); under topics including description of use, consequences and relations with other patterns. For example, *Producer-Consumer* is a key pattern that Björk and Holopainen discuss in relation to Pac-Man, as this pattern determines the availability and life-time of *Resources*. They note that the pattern thus governs the flow of gameplay, as play progression is determined by consumption of Pills. It is nevertheless a very simple pattern in Pac-Man, compared with later games: a fixed number of Pills (and Power Pills) are produced only at the start of each level. A bonus item (Cherry) is produced at pseudo-random intervals, but over the course of a long game its impact is negligible. *Producer-Consumer* in Pac-Man is also simple because it consists of only a single pair[2] of *Producer*—the new game levels—and *Consumer*—the Pac-Man. Björk and Holopainen (2005) define the general *Producer–Consumer* relations as below; we indicate the subset of these relevant for Pac-Man by striking out those which do not fit:

- Instantiates: *Varied Gameplay*, *Resource Management*
- Modulates: *Resources*, ~~*Right Level of Complexity*~~, *Investments*, ~~*Units*~~
- Instantiated by: *Producers*, *Consumers*, ~~*Converters*~~
- Modulated by: ~~*Container*~~
  Potentially Conflicting: ~~*Illusions of Influence*~~, *Predictable Consequences*

It may be instructive to see how related patterns instantiate in Pac-Man. *Varied Gameplay* is derived by the use of Power Pills—with these, the modes and roles of the game switch in possibly strategic ways. *Resource Management* refers primarily to Pac-Man's lives: since the game cannot be won, but only played for high scores, the preservation of a buffer of extra lives is vital to longevity. *Resources*, as mentioned, are the Pills and Power Pills, and occasionally bonus Cherry. *Investments* are instantiated as tactical use of the Power Pills; while some might use the Pills to hunt Ghosts and gain points, others might use them only to buy time to clear the level more easily, minimizing risk. *Producers* and*Consumers* area as described above; finally *Predictable Consequences* occur because whatever consequences emerge from these patterns remain fixed throughout the game, even if all events speed up over levels.

This pattern description of Pac-Man is not guaranteed to be complete, merely illustrative. In practical applications, the pattern description needs only to cover the player's agency, and only to the degree that player behaviour will be well mapped to Behavlet candidates. After that the constraints met while building and validating the Behavlets will begin to shape the outcome.

4.2 Feature/Behavlet identification

To apply the theory from our approach, it must be related to the game. This is the point at which expert knowledge becomes important—to make the leap from theoretic to instantiated trait, it is mostly necessary to know the game and translate the wording of the trait definition into the terms of the game.

[2] Where the pattern in other games can consist of many combinations of pairs or even many-to-one or many-to-many relationship structures.

Table 7 The subset of behavioural traits from Table 3 which can be instantiated in Pac-Man, with descriptions

Behaviour trait	Description for Pac-Man
Caution	In Pac-Man includes, e.g. how much distance the player *consistently* keeps between Pac-Man and the ghosts when not in the Hunting mode
Thoroughness	In Pac-Man is defined by the proportion of each level's absolute maximum score which is achieved by the player
Aggression	In Pac-Man relates to how consistently and riskily a player chases after the ghosts after eating a Power Pill
Decisiveness	In Pac-Man reflects in how much or little time is spent without making gains
Planning	In Pac-Man includes, e.g. luring Ghosts near to a Power Pill and then hunting, eating all Ghosts
Optimisation[a]	Is the key motivator in Pac-Man: success in the game is defined in terms of high score comparison between players
Resourcing	In Pac-Man means managing lives, fruit and Power Pills (which are often hoarded during a level's play to be deployed at strategic moments)
Speed	In Pac-Man reflects primarily in how fast players clear each level
Control Skill	In Pac-Man is most easily indexed by how well the player deals with Ghost near-misses, which require good skill to escape at higher speeds

[a] High-score related Behavlets can be observed only when many games are played

4.2.1 Traits of Pac-Man play behaviour

The list below in Table 7 illustrates the idea of instantiated traits. The list retains enough of our play-traits to comprehensively cover Pac-Man but is not suggested to be complete for all games: since there is no multiplayer or anthropomorphism in Pac-Man, empathy and morality do not appear.

The point of this list is to help focus on specific behaviours in Pac-Man, so that the search for features is both guided and bounded; in game-play analysis the features can also help to relate observation to theory.

4.2.2 Traits versus patterns in Pac-Man

Here we simply match the patterns identified for Pac-Man with associated archetypes, behaviours and preferences, see Table 8. We select only from the set of patterns listed above in Table 5 for the sake of clarity for the reader; of course all tables herein are for illustrative purposes and should not be used as reference sets when applying the method.

4.2.3 Pac-Man observation and Behavlets

We proceed to develop a set of Behavlet concepts for Pac-Man, each relating to one or more traits. This has an observational component, including the game analysis given above and watching Pac-Man games being played-searching for and registering

Table 8 Association of play patterns and archetypes, linked through the behaviours and preferences of the type

Archetype	Behaves with…	Archetype preferences	Associated game design patterns
Logistical	Thoroughness	Optimisation	Lives, High Score Lists, Pick-Ups, Geometric Rewards for Investments
	Caution	Planning	Time Limit, Producer-Consumer, Resources
	Resourcing		Movement
Tactical	Aggression	Improvisation	Enemies, Inaccessible Areas
	Speed	Thinking on the spot	Enemies, Time Limit, Inaccessible Areas
	Control Skill		Collection, Movement
	Resourcing	Operation	Levels, Producer–Consumer
	Speed		Collection, Movement
	Decisiveness	Controlling single characters	Enemies
	Speed		Collection, Movement
	Control Skill		
Strategic	Optimisation	Solving	Geometric Rewards for Investments, Resources
	Resourcing		
	Planning	Thinking ahead	Enemies, Role Reversal, Resources
	Speed		Collection, Movement

distinct patterns of events. The intent of the player in performing each action sequence is associated with a behaviour trait.

To observe game play and explore the game log, it can help to use data visualisation to illustrate interactions between game system and player behaviour. For instance Fig. 4 plots the distance between Pac-Man and Ghosts (showing only two for clarity), alongside the record of performance (Points and Pills, scaled to fit by constant factors), and resource use (Power Pills and Lives), giving a picture of player caution and suggesting possible Behavlets.

Decrements in the *Power Pills* line show where Pac-Man has eaten a Power Pill and 'flips' the play mode. The Power Pills are eaten close to points where the Pac-Man to Ghost distance hits zero, while Pac-Man's Lives do not decrease. Together this suggests that the player prefers to use Power Pills to hunt Ghosts rather than focusing only on clearing the level. Comparing the two Ghost-distance traces at these Hunt-mode points shows that distance for one Ghost tends to increase while the other is decreasing, reflecting how Pac-Man is chasing one Ghost while the other is driven to escape. In the case where Pac-Man catches both Ghosts, around state 250–270, Ghost 1 gets quite far until Pac-Man has caught Ghost 2. This indicates that the player is

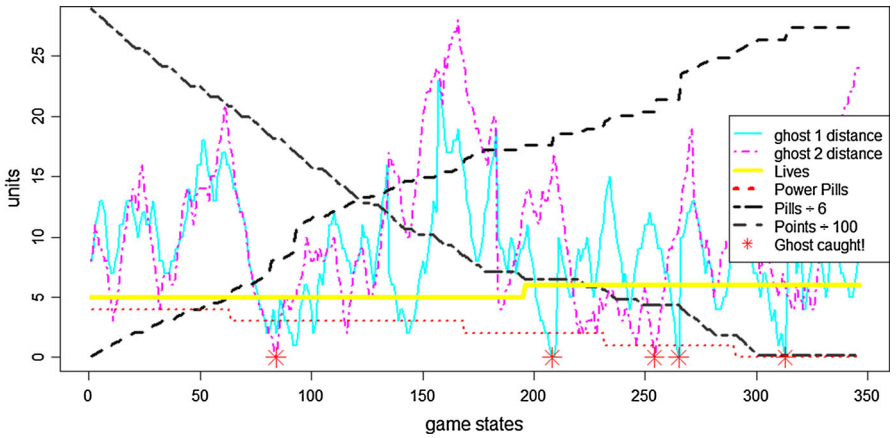


Fig. 4 A time series depiction of a single game level. The distance values between Pac-Man and two Ghosts are overlaid with simple metrics of gameplay, in some cases scaled by an arbitrary factor in order to display

not arranging these episodes efficiently by waiting until more Ghosts are close before triggering the Power Pill effect.

One Behavlet suggested by this kind of analysis was *C3_Close Calls*, with the natural language description: *the number of times Pac-Man comes within one square of a Ghost when not in the Hunt state, and is not killed while so close*. It is clear that this Behavlet will be feasible to describe with a logical constraint harness. It is also reasonable to expect the Behavlet to generate data; it is not an atomic action; it is probably not related to any other trait than *Caution*.

In a relatively less-structured domain, such case-by-case behavioural insights would be very difficult to generalise, but with design patterns regularising the domain of game-play we can be more confident that observed behaviours reflect genuine patterns of engagement. Such micro-detailed examination is complemented by a higher-order view; for example, a common technique in game quality testing is to examine the heatmap of player (and player's opponent) activity in a map or level, as in Fig. 5. The map for Pac-Man's movement illustrates that this player is quite aggressive, having a mid to high count of moves across the front of the Ghost house. We can also see from the Ghost heatmap that the player was not very threatened, as they did not cover most of the map's right hand portion.

Another approach is to describe obvious game subsystems, such as Ghost behaviour in Pac-Man, using a finite state machine (FSM). By their nature, FSMs permit analysis at any level of abstraction. Adapted from Schell (2008), Fig. 6 shows such a FSM which illustrates that Ghost behaviour can be represented quite simply. This FSM could be used to generate the Markov model of Ghost behaviour by assigning transition probabilities, based on the combination of Ghost probabilities defined in the game engine, and player probabilities extracted from sample games. The resulting model is a useful tool for determining the relative likelihood, and thus subjective importance, of different player–Ghost interactions. Such information may then suggest further Behavlets.

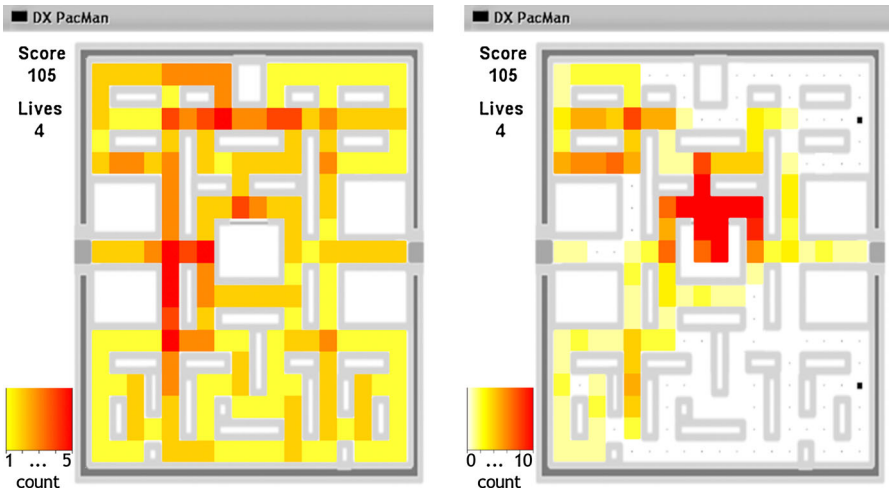


Fig. 5 Heatmap of the movement over one level of (*left panel*) Pac-Man, and (*right panel*) two Ghosts (level screenshots colour-inverted, level data as per Fig. 4) *Colour-bars* indicate the count of times each map square is traversed, with the range 1–5 (*left panel*) and the range 0–10 (*right panel*). (Color figure online)

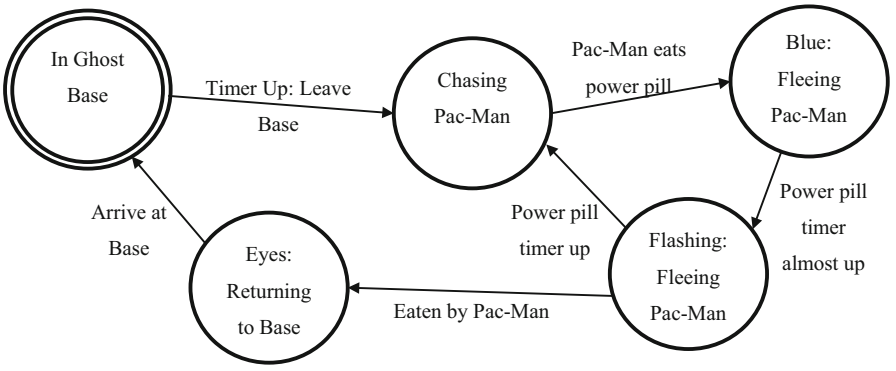


Fig. 6 A simple state machine of Ghost behaviour, adapted from Schell (2008), with permission

One example Behavlet associated with each Pac-Man trait is listed in Table 9; excluding the trait *Optimisation*. Although this is a key trait for Pac-Man, it is only expressed when many games are played. Behavlets which could capture multi-game play behaviour would require a persistent player model, which was outside the scope of our focus during this analysis. The complete set of Behavlets for Pac-Man is described in this manner in Appendix B in Supplementary Material.

4.2.4 A note on Behavlets versus features

To illustrate the distinction between Behavlets and the type of feature more generally used, we define some of these 'simple' features for Pac-Man in Table 10. Simple

Table 9 List of Behavlet concepts after Pac-Man game observation, arranged by trait association in column order

Pac-Man trait	Behavlet name	Natural language description
Caution	C1_Times Trapped By Ghosts	Threat perception: Pac-Man trapped in corridor by Ghosts—i.e. one ghost on each possible direction of movement (vector)
Thoroughness	T1_Sector by Sector	Clear Pills by sector, not moving on until sector is completely clear
Aggression	A1_Hunt Close To Ghost House	Counts instances where Pac-Man follows the Ghosts right up to their house while attacking them
Decisiveness	D1_Pac-Man Vacillating	Oscillating movement with no ghosts near
Planning	P1_Lure Ghosts to Power Pill	How often the player waits beside a Power Pill to lure the Ghosts in
Resourcing	R1_Average Time For Pac-Man to Eat Cherry	How quickly the player collects newly appeared Cherries on average
Speed	Sp1_Average Cycles Per Sector	Count how many cycles it takes the player to clear a given sector
Control Skill	CS1_Teleport Use	Count instances when Pac-Man uses teleporters to escape a threat

Table 10 Features which count simple aspects of play

Simple features	Description
Levels	Count the number of levels played during the game
Lives Gained	Count number of times Pac-Man gained lives, i.e. ate a Cherry
Lives Lost	Count number of times Pac-Man lost lives, i.e. caught by ghost
Distance Pac-Man to Ghost *i*	Pac-Man's distance from Ghost *i*
Cycle Count Per State	Number of Cycles per Pac-Man move
Points Max	Maximum Points value obtained, i.e. score from the game
Cherry Onscreen Time	Total number of states during which the Cherry was onscreen

For re-occurring events, e.g. *Cycle Count per State*, there will actually be as many features as statistical models that can be fit to the data, e.g.: mean, minimum, maximum, standard deviation, etc

features require no inference or induction; they are simply counts or results for some 'basic' operation on interesting variables from the raw data. However many of the Behavlets rely on the simple features to derive their results. These simple features may be no less important in characterising behaviour than the Behavlets, but this is an issue to decide during feature selection.

For a larger game, the instantiation of traits and patterns will require an iterative process; with each iteration increasing specificity. Each list will also help to refine the other, as design patterns show the available actions and behaviour traits illustrate the likely activity.

4.3 Pacman Behavlet coding

Finally, at the third stage of the process, an algorithm encoding is described to represent the Behavlet formalism and capture the observed behaviour. We start with the Behavlet concept, and from first principles create a specification for the concept—thus reasoning about the specification of a Behavlet inspires the algorithm which derives that Behavlet.

The coding methodology calls for development of an algorithm to represent the Behavlet's natural language description, specifying in pseudo-code a constraint harness with parameters, and a consequent measurement function to return a cumulative variable for the Behavlet. The constraint harness is a set of logic statements, and the parameters are simply numeric thresholds. An example is the initial constraint for the first Aggression feature, *A1_Hunt Close To Ghost House*, written in Structured English:

```
IF play mode EQUAL power pill AND all ghosts distance to ghost house LT 3
ENDIF
```

This constraint triggers when Pac-Man is hunting the Ghosts and at least one Ghost is a Manhattan distance of 3 or less from their House. Within the 'then' clause there is more logic to induce whether Pac-Man is closing in on the House, and what happens when he gets there. If all the conditions are satisfied then the Behavlet function

Table 11 Sample of Pac-Man Behavlets, which model a player's behavioural preferences

Pac-Man Trait	**Behavlet name**
Structured English Pseudo-code Description	
Caution	C1_Times Trapped By Ghosts
CREATE two outputs DO increment game state IF play mode EQUAL normal AND ghosts block all A* paths from Pac-Man to opposite point on map increment first output IF Pac-Man lives > Pac-Man lives after 10 game states increment second output ENDIF ENDIF UNTIL last game state	
Thoroughness	T1_Sector by Sector

CREATE one output
divide map into 3x3 grid
DO increment game state
GET Pac-Man location in grid
WHILE Pac-Man stays at grid location
increment game state
ENDWHILE
IF no Pills remain in grid
increment output
ENDIF
UNTIL last game state	

Aggression	A1_Hunt Close To Ghost House

CREATE one output
DO increment game state
IF play mode EQUAL power pill AND all ghosts distance to ghost house LT 3
WHILE play mode EQUAL power pill
increment game state
IF Pac-Man distance to ghost house LT 3
increment output
ENDWHILE
ENDIF
ENDWHILE
ENDIF
UNTIL last game state	

Column 1 refers to the associated trait; Column 2 contains the Behavlet name; Column 3 (underneath columns 1 and 2) gives a pseudo-code description of the Behavlet 'constraint harness'. Showing only the first three traits, for brevity; remainder are reproduced in Appendix A in Supplementary Material

increments a counter, which will be the final result after iteration through all the raw data.

Table 11 re-lists the sample of Behavlets introduced above (Table 9), with descriptions in pseudo-code to illustrate the relevant operations on the game log variables.

Each piece of pseudo-code was implemented in our C++/DirectX Pac-Man engine, as the logic at the core of the function that derived the Behavlet. Some functions look at a wider spread of the raw data than others, and some try to meet more complicated constraints, but the algorithmic pattern remains much the same throughout.

Not every Behavlet concept will result in a usable implementation, due to the nature of the coding task being performed. For instance, reliable results could not be returned for the proposed Behavlet *P5_Pausing*, due to the complexity of trying to capture the intentionality of players in a logical predicate about the rhythm of play. Such Behavlets should be discovered while writing pseudo-code.

4.4 Feature selection

When preparing to create a model from Behavlets and/or features, feature selection for 'quality assurance' is normally performed to find potentially useful features, and discard the rest. It is possible to degrade the performance of some algorithms by using too many features, and maintaining large feature sets in an applied setting can create overhead. Any 'Feature Selection' module operates relative to some target of learning (e.g. a player type label); and should provide two basic functions. *Screening* removes unimportant and problematic predictors, such as predictors with too many missing values or predictors with too much or too little variation to be useful. *Ranking* sorts remaining predictors and assigns ranks based on the predictive importance to the target to be learned.

Lastly, as a simple 'sanity check', we built a correlation matrix with threshold of 0.95, to determine which Behavlets/features returned redundant information. For instance in Pac-Man, the number of levels played corresponds highly with number of Ghost Hunts, because the number of Hunts per level is fixed by the number of Power Pills.

5 Results—evaluation

5.1 Prior work

Classification and prediction are two notable uses of features or Behavlets. We have previously published papers on both of these topics, using behavioural features of Pac-Man, though without using the term 'Behavlets' as such. These features were similar to the Behavlets just described, but were 'naïve' in the sense that they did not have the same grounding in theory that Behavlets do. In Cowley et al. (2013) we presented a study of player type classification, which illustrates how Behavlet-like features can generate useful insights for the data miner. However that study does not provide a controlled comparison of the use versus non-use of such features. In Cowley et al. (2009) a controlled comparison is made between two Decision Theory models for predicting future player moves, one using simple metrics of gameplay and one using Behavlet-like features. The latter model improves speed (from non-real-time to real-time), and accuracy by 35 %.

5.2 Evaluation workshop

To compliment this literature, we conducted an evaluation workshop with a sample from the target audience population: game designers. The target game was Gears of War, chosen because it represents a modern, sufficiently-complex example of a familiar genre, comparable to other games from the genre, and completely satisfies the reasoning condition.

5.2.1 Participants

Nine participants attended, recruited from a computer science research institute and a game development studio. The group included seven software developers and two games researchers. All had game development experience, and all but one were regular players and experienced gamers.

5.2.2 Procedure

The workshop lasted two and a half hours, and was moderated by the second author. The moderator first gave an overview briefing on: temperaments and traits (as described above); game design patterns; the concept and rationale of Behavlets. The complete list of Björk and Holopainen (2005)'s game design patterns were distributed as printed cards for reference.

The workshop duration was fixed by reference to meeting duration effectiveness reported in Romano et al. (2001); total working time lasted 2 h.

The procedure required participants to work in pairs. Each pair was given a document consisting of a table with the 11 traits defined in Table 4. The table had two empty columns: one for design patterns and one for the Behavlet concept: i.e. a description of behaviour in FPS games which should map from the trait to any of the design patterns. Participants were asked to assign design patterns to traits, and map between them with Behavlet concepts: the ideas for behaviour-related features of gameplay to help distinguish between types of player. The session was partially interactive—there were occasional general discussions and points of clarification to the group (every ∼15 min). The moderator took notes on all discussions for reference.

Participants had all completed Gears of War and were encouraged to focus on this game for the design exercise. However they were also to consider common features of play from other similar "action/shooter" games such as Halo, Call of Duty and Destiny, to help complete a full set of Behavlets.

5.2.3 Method of analysis

The moderator collated responses and, working against the discussion notes, checked for omissions, typos, and duplicates. Duplicate design patterns were counted and merged. Duplicate Behavlets were re-worded and collated. Discussed Behavlets omitted from the participants' own notes were added if necessary.

5.2.4 Results

A partial list of the workshop output is reproduced in Table 12, with Behavlets alongside their matched traits and design patterns (only for the first trait, Caution; all other traits are covered in Appendix C in Supplementary Material). These tables thus provide a rough mapping from traits to design patterns, ready for refinement by observation and coding.

There were 86 unique design patterns assigned to the 11 behaviour traits defined above in Table 4. 31 of these patterns were assigned to separate traits multiple times, resulting in 57 duplicates and a total count of 143 patterns. The number of patterns that each trait received, and number of duplicates of each pattern, is listed in the tables

139 Behavlet concepts were conceived in the workshop, mapping from the 11 behaviour traits to the 143 design patterns. Many Behavlets overlapped, aiming to capture the same behaviour, and the count and reworded list of these is included below. Behavlets are subjective, therefore the participants' original wording is retained in Table 12 and Appendix C in Supplementary Material. The list below is the moderator's interpretation of the overlapping behaviour, with traits matched to Behavlets and a count (out of nine) of participants who made the same/similar contributions.

Caution	Staying out of range and line of sight of enemies, and blind firing	4
	Preferring long range weapons	3
	Moving as a group	2
	Peeking round corners—being careful of open areas	2
	Use of Sniper rifle scope or binoculars to scout area	2
	Takes cover early to regain health	2
	Frequent saving	2
	Camping	2
Meticulousness	Attempt to complete all tasks, gain all achievements	6
	Collecting items/weapons	4
	Area coverage	3
	Area clearing	2
	Easter egg collecting/uncovering exploits	2
	Play precision (precise kills etc.), optimizing kit	2
Impulsiveness	Aggressive behaviour (open frequent attacks, less planning/strategy)	6
	Weapon Choice	4
	Less mindful of cover	2
	Less cautious about health status	2
Competence	Optimized, real-time decision making (e.g. weapon selection, motion)	4
	Skill matching and optimized planning (players and quests)	4
	Score /accuracy 3	3
	Practice/frequent play	2
Logic	Equipment selection	5
	Quest/path planning (including team)	4
	Battles (actions expose knowledge and planning)	2
Perfectionism	Optimized equipment section/set up	4
	Perfectionism and repetition (zero deaths, ideal combat stats)	3
	Completion-ism (tasks, equipment, achievements, items)	2
Empathy	Playing a support role (reviving, team work)	4
	Teamwork	3

Morality	Team ethic (e.g. sharing loot, not taking kills)	4
	Not camping or preying on the weak or showboating	3
	Not being disrespectful (e.g. not mocking via gestures, no abusive chat)	2
Resourcing	Item/weapon searching/collection	7
	Managing resources (e.g. in menu system)	2
Speed	Completion rate	5
	Points/attempts ratio	2
	Weapon selection and accuracy	2
Control Skill	Ability/weapon choice (particularly real-time)	2
	Precise weapon use	2
	Player character motion (particularly in battle)	2
	Shooting accuracy	2

The subjective responses to the workshop were generally positive. Participants expressed no issue in understanding the basic concept of Behavlets. Several participants gave feedback on the process. Topic headers 'understanding', 'usefulness', and 'improvements' were suggested, though not enforced:

Game researchers

1. Understanding: *The explanation of the process was described clearly*. Improvements: *Using the game mechanics notes was relatively time consuming because of the detail given. A suggestion would be to give a shortened version of the game mechanics with a short explanation and if required more detail can be searched for through the detailed notes*.
2. Understanding: *The basic system was very easy to understand and use as the example of Gears of War was very well known and had many actions within it that could be translated across to behavioural traits*. Usefulness: *The workshop hand-outs helped explain each trait in detail and from these it was easy to breakdown the gameplay traits of Gears of War*. Improvements: *I can't suggest any improvements as the hand-outs coupled with the well-known example made for an interesting and easy to understand workshop*.

Game developers

1. *Using the system as part of a group was a very interesting and insightful process. Whilst all the group members enjoy playing GOW, they all display distinctly different playing behaviour and have a diverse range of personality types. By using the system it was easy for each of us to examine our on playing tactics and style and map these to design patterns. It was interesting to find out that the many game design patterns were suited different types of people.*
   We all enjoy playing GOW but it was clear from this investigation that we all use different designs patterns that directly suit out personality type and player behaviour.
2. Understanding: *Process was clearly explained and easy to understand but the quantity of design patterns was hard to absorb within the time given.* Usefulness: *Breaking down the gameplay traits of a game was useful as it provided direction to the analysis of our play styles*. Improvements: *Some traits were difficult to map to design patterns. Especially when it came to traits similar to "The player does not*

Table 12 Behavlet Identification (Gears of War—with elements of Halo/Call of Duty/Destiny)

Original	Applied	Behavlet concept, mapping gameplay behaviour trait to design patterns	Design Patterns in FPS games, e.g., Gears of War	Duplicates
Caution (26 patterns)	Caution: watchfulness, alertness	Difficulty choice	Camping	4
		Keeping allies alive	Area control	3
		Frequent saves	Evade	3
		Focus on trigger or control points (in Domination mode)	Manoeuvring	3
		Move as a group	Survive	3
		Choice of weapon	No ops	2
		Prefer long range weapons—away from danger	Safe haven	2
		Keeping a high Kill/Death ratio	Save load cycles	2
		Preferring third person cameras with melee weapons (e.g. Destiny)	Stealth	2
		Changing weapon to one with more ammo—even if less experienced with the new weapon	Strategic locations	2
		Clearing area before collecting items	Ability losses	1
		Uses long range weapons more often	Aim and Shoot	1
		Checks around corners carefully	Collecting	1
		Moves in crouch—especially noticeable if in safe areas	Damage	1

Table 12 continued

Original	Applied	Behavlet concept, mapping gameplay behaviour trait to design patterns	Design Patterns in FPS games, e.g., Gears of War	Duplicates
Caution (26 patterns)	Caution: watchfulness, alertness	Difficulty choice	Camping	4
		Stays clear of open areas or moves through quickly	Downtime	1
		Staying out of enemy sight	Enemies	1
		Camping	Hovering	1
		Frugal with special weapons ammo (e.g. grenades)	Limited set of actions	1
		Pays close attention to game cues (audio, graphics, map)—hesitant, stopping	Penalties	1
		Use of scopes and binoculars to check areas	Pick ups	1
		Take cover or run away early after being hit to recover (even when health still good)	Power ups	1
		Frequent reloads—even when weapon full	Reconnaissance	1
		Hiding	Renewable resources	1
		Running away	Resource management	1
		Remaining in a location or staying still	Savepoints	1
			Spawn points	1

First two columns show the Temperament type behaviour trait, with number of assigned design patterns. Third column shows the mapping between the trait and the design patterns, listed in column four, which were observed from gameplay by the participants. Fifth column shows the number of duplicated design patterns, i.e. number of participants (out of nine) who made the same assignment to this trait

Showing the first trait only, for brevity; remainder are reproduced in Appendix C in Supplementary Material

take cover very often". It was not clear how you should map traits describing the absence of a behaviour to a design pattern. Perhaps the procedure for this should be overtly defined or incorporated into the system somehow.

3. Understanding: *The process was easily understood and was complemented well with the material provided. The game chosen for reference was good as everyone had heard of it or player it, however it was sometimes easier to make references to other games (Destiny came up quite a lot). The material provided explained the game play traits and behaviours well which helped when making the connections between them*. Improvements: *As avid gamers we had a lot of input and probably could have used more time other than that I can't make any suggestions*.
4. Understanding: *The system was quite simple to understand and Gears of War is one of my favourite games there was many actions within it that could be translated across to behavioural traits.* Usefulness: *With the hand-outs examples it was easy to see the traits that were in Gears of War, I could also distinguish the type of player and personality I am from this workshop*. Improvements: *Nothing springs to mind for this, I mean the hand-out explained what was needed and as it was Gears of War, well there aren't many better examples*.

Based on these results and on the moderator's reflection, the workshop is discussed in the next subsection.

5.2.5 Workshop discussion

The workshop outcomes would have benefitted from a longer duration, possibly four to four and a half hours for a thorough analysis and mapping. On the other hand, it was difficult to maintain focus over the period, and it might help to limit the work on each trait area to about 20 min. Another session (or 'hackathon') is needed to identify Behavlets in the game engine by observation and coding; and after that there still remains the very practical work of feature set shaping and optimisation to recording in real-time.

The sheer number of design patterns to deal with causes a cognitive challenge of trying to keep many concepts in mind. However many of the Behavlet ideas are 'obvious', given a sufficiently expert designer, and so this is less of a problem than it is an opportunity for improving the process by adding supporting tools. Even more traits and design patterns could be identified—particularly around story, social and meta-game situations. However, the first and key step is for designers to discuss core gameplay, and the gameplay traits—either positive or negative—that they expect players to exhibit, separated into temperament groups.

The fact that a substantial number of duplicate Behavlet concepts were produced suggests that (a) the traits and design patterns (that served as input) were commonly understood, and (b) the process has general validity. Given that the lists are the result of aggregating a group process, it is clear that there should be overlap between separately working participants, and potentially even disagreement. We consider it beyond the scope of the method itself to define how artefacts of the implementation process such as this should be resolved. Simply adding additional rounds of debate, aggregation and refinement, to a similar workshop structure, could be sufficient. On the other hand, the aggregate list produced by the authors above illustrates how the initial material

of a next round could appear. Duplication count provides a clear metric for focusing the time-consuming steps of Behavlet observation and coding. A system with such a core of obvious, operational Behavlets can be easily grown because the game atoms observed and coded can usually be reused in more non-obvious Behavlets that did not get duplicated.

In this context the traits Perfectionism and Meticulousness proved to be only subtly different, and so tricky to separate the resulting Behavlet concepts. These traits could potentially benefit from revised wording for better separation in general application. In this format, the process needs to be led by someone familiar with it, and Behavlet mapping will need to be tidied up after responses are returned. However these are issues of the kind that should disappear with practice. The workshop was very successful considering participants were unfamiliar with the approach.

6 Discussion

6.1 Behavlets and other approaches

A major focus of player modelling research has been *game metrics*, i.e. using simple logs of game engine variables to extract statistical features of play, and cluster or classify players (Drachen et al. 2009). Creation of game metrics, by extracting features from raw data, can be thought of as an operation of dimensionality reduction. One optimisation principle to derive and select features is to minimise the cost of effort. On the other hand, at the minimum effort-cost the selection of features will be arbitrary, which prevents hypothesis making. One solution is to work from a theoretical model to either derive or interpret features. All modelling approaches require the two steps of derivation and interpretation. Manual input is required for at least one of the steps, i.e. the investigators must make explicit judgements on a case-by-case basis either to design features from theory, or to interpret the outcome of modelled features with respect to theory.

The majority of approaches, in the field of computational intelligence in games, tend to use straightforward features and try to link them to theory during interpretation. Normally, such features contain information only in their (combined) distribution. Compared with a quantitative theoretical construct, such as Intelligence Quotient or 'IQ' testing, this is much less informative of the background that the theory describes. Hence further work is necessary to interpret the machine learning output. Therefore, we argue that the Behavlet method does not add complexity or effort, but rather serves to shift the timing of when the effort occurs. Behavlets represent a novel way to meet a pre-existing need to engage with the theory of behaviour. With the structural relationship established by the Behavlet method, observed behaviour can be interpreted in widely-accepted psychological terms, which also allow normative comparison between players.

Other work has focused on extracting maximum information about player behaviour from simple metrics, e.g. Drachen et al. (2009). Compared to our approach, this method seems to only translate the main work from extraction to interpretation, and additionally does not build on established psychological theories. On the other

hand, the core concept could potentially be complementary to ours, if we consider an iterative approach shuttling between each method in turn. This combination would fit neatly into the established iterative feature-extraction paradigm our method uses: a domain expert's top-down knowledge motivates observational experiments in the problem space, and the bottom-up information gained from these experiments helps to advance understanding in the next iteration.

6.2 Limitations

6.2.1 Complexity

When applied to a complete and novel game, the duration and difficulty of the Behavlet process tends to be proportional to the complexity of the game, for two main reasons. First, it is initially time-consuming but necessary to lay an analytic foundation in step 1.c with the game design patterns method. Second, the observation of gameplay in step 2.c is open-ended because, short of formal modelling, only direct experience can support a judgment that enough examples of behaviour have been observed and documented. Thus the worst-case Behavlet process would involve on the order of $2\times$ 'gameplay completion time', that is, the time to complete all possible novel gameplays. However, we are justified to disregard this worst-case scenario by the following.

For the first reason, step 1.c and the issue of mapping patterns, by their nature game design patterns must be repeated across games and thus even novel games tend to present familiar elements. Also, there is a large and growing literature of design patterns already published for games of many genres, which should be straightforward to adapt. For the second reason, step 2.c and the observation and search for behaviours, one simple aid is to use a clustering algorithm to identify which simple player actions tend to co-occur. This approach was reported in Cowley et al. (2014), where the event clusters were also associated with design patterns (although no Behavlet analysis was conducted). Such clusters should then speed up discovery of patterns of play, and help in associating them with temperaments.

Finally as a general point, this method is intended for application in a context where extensive analysis of the game should be performed anyway, such as during development or research. Participants in our workshop are comparable in expertise to anyone researching games, and to the extent that the Behavlet method was explored there, the more complicated games used did not seem in incur a major increase in complexity.

There remains the valid question of whether there is a tractability 'threshold', games with so much inherent complexity that even using machine learning 'short-cuts', it would remain unfeasible to apply the Behavlet method. Open world games such as the Grand Theft Auto series might be of this variety, because they can be played with as an unstructured toy, where the only constraints are players' creativity. Given that completely unconstrained toy-play does not meet our 'reasoning condition', it is clear that such games are simply invalid candidates for the Behavlet method. However, a subset of the games, constrained by e.g. mission parameters, could easily be valid. In a similar way, we would argue that massively multiplayer online games (MMOGs)

are valid, because the *social* constraints are sufficient to limit most players' behaviour. This view is founded on the work of Bartle (1996), who found clear behaviour patterns in MMOGs, and Stewart (2011) who linked those patterns to temperament theory.

6.2.2 Linking temperament theory to observed behaviour

A separate question of validity is whether there truly exists a causal link from temperament theory to observed behaviour. Can we claim that temperament significantly influences play behaviour, and if so, is it stable across time?

Firstly, given the wealth of evidence (for example, Bartle 1996; Drachen et al. 2009) that finds constrained variation in playing styles, i.e. player *types*, it is clear that there is some factor which drives players to distinguish themselves. It is not clear what this factor is, in general psychology any more than games; many theories exist and are not all mutually exclusive. However the theories we draw on have been empirically tested to show some explanatory power, and even earlier versions of our work (Cowley et al. 2013) have found clear relationships between player type labels and observed behaviours.

In the question of stability, it can be reasonably expected that the strict answer is no. In Cowley et al. (2013, Sect. 5.2.2, p. 518) we discussed the issue of concept drift and showed how the classification of player type drifted in response to player learning. The picture from that data was that learning tended to evolve playing styles toward a common mean, which might represent some optimal strategy of play inherent in the game design itself, and thus unrelated to player psychology. It is reasonable to expect that the more complex a game is, the more playing time it will take to learn such optimal strategies, and thus the more scope there is for personal expression. For sufficiently complex games, there might be no such constraints.

Finally, it is worth noting that the temperaments are only ever used as an approximate model. Given the probable limited resources available to exploit the Behavlet player models, there is anyway no use case for hyper-detailed models. For building most sorts of adaptive functionality, clustering players into four types should be sufficient.

6.2.3 Generalisation

The potential limitations above are also relevant to the generalisation of the Behavlet method beyond games, that is, in a *user modelling* rather than *player modelling* use case. As discussed, context is provided by the game when doing player modelling, and the psychological embedding of the player in the game is a reliable consequence of Huizinga's classic *magic circle* concept (1949). For user modelling in other domains, assumptions about the user psychology must be built on other theories, which potentially makes it harder to specify Behavlets. However the approach does not change, only the necessary groundwork. Aside from this, nothing in particular ties the Behavlet method to games. For example, in any behaviour change context—e.g. weight loss—patterns of behaviour that lead to eating inappropriately or avoiding exercise can be modelled, and psychological frameworks of explanation are well defined. The outcome could possibly be used to adapt the weight loss intervention online. In active rehabilitation—e.g. exercise promoting falls prevention for older people or upper arm

physical rehabilitation after brain injury—the key goal is to boost engagement and motivation, which are intrinsically linked to personality. This is a prime use case for gamified interventions linked reliably to psychological theory.

6.3 Future work

We believe that there is more to be gained from the Behavlet approach, predicated on two main additions to the method: formal notation of the game space, and enhancement of Behavlet specifications by machine learning. We advocate the use of formal methods to describe the game space, a methodology which can help the domain expert to extract features of play defined in terms of the available action sequences in the game (Breining et al. 2011); this helps to marry formal modelling with the experts' informal domain knowledge.

In addition to easier discovery of Behavlets via event clustering, machine leaning can be used to perform information fusion within the set of all features and Behavlets. For example, game design patterns tend to occur both over short durations, e.g. within a level, and over long durations, e.g. over several levels. Interesting work by Kludas (2011) shows how information from disparate sources can be used to construct meta- or compound features, which could combine Behavlets at such lower and higher levels of game-play. Building on this, we envisage creating complete player personas which take into account the variation in playing styles across multiple levels of the game. Another use of machine learning is to imbue artificial players with human-like 'character' by drawing on the Behavlets, specifically reverse-engineering the 'constraint harness' of a Behavlet to create a 'human-like' play pattern script in the game engine. An agent controller could then drive certain usage patterns of this script to conform to particular player personas, with similar benefits as described in Holmgard et al. (2014).

Of course the best 'proof of principle' will be to use the Behavlet approach during funded development of a game by professional developers. Using the method in a practical setting would road-test the implication that Behavlets provide a structured way to model personality in a game, whether as a player model or non-player character. The other papers in the special edition deal with the importance of this—particularly Harley et al. (paper 961, *this special issue*, 2016) in relation to behavioural change contexts, e.g. exercise.

Following such future work, we envision that Behavlets can lead to improved games by supporting a richer toolset for the designer to understand and develop their game's capabilities and agencies. Specifically, the activities of play that arise in a game can be well-defined and mapped. Also, the reactions of players to the game can be understood in a unified, theoretically-sound way. Both improvements lead to greater insight.

6.4 Conclusion

Feature engineering can be approached in an iterative two-step process. A top-down understanding of the problem domain, and the strengths and limitations of the chosen classifier, motivates candidate solutions and experiments to test them. Bottom-up information gathered in experimental work tests assumptions and drives a new iteration

to converge on a working solution. Our method fits this two-step process into a number of extant theories of game playing. Analytic observation is used to propose candidate Behavlets, by finding and observing patterns in game-play, and deconstructing their operation to the atomic level of state transitions to be codified as an algorithm. The method enables the characterisation of playing styles by creating observable indicators of player variation. This costs more upfront effort to frontload richness of description into the feature space, but the payoff is an easier and less ambiguous interpretation step after clustering. Though our approach is potentially more time consuming we contend that it has the clear benefit of developing more meaningful and interpretable models of users, with the potential to capture the way players themselves approach game play.

Finally, when theory drives feature creation, there are clear benefits. The classical hypothesis testing model is valid, even allowing actual behaviour predictions. The link from theoretical models to observations is less ambiguous. The chance is increased that a feature's action sequence is a good classifier in theory space, i.e. the space of the actual models that describe human psychology.

Acknowledgments This work was partly supported by Tekes the Finnish Funding Agency for Innovation (project Re:Know 5159/31/2014).

References

Acuña, D.E., Parada, V.: People efficiently explore the solution space of the computationally intractable traveling salesman problem to find near-optimal tours. PLoS ONE **5**(7), e11685 (2010)

Alexander, C.: A Pattern Language. Fachhochsch, Fachbereich Architektur, München (1990)

Bakkes, S., Tan, C.T., Pisan, Y.: Personalised gaming. In: Proceedings of The 8th Australasian Conference on Interactive Entertainment Playing the System (IE '12), pp. 1–10. ACM Press, New York (2012). doi:10.1145/2336727.2336731

Bartle, R.: Hearts, clubs, diamonds, spades: players who suit MUDs. J. Virtual Environ. **1**(1) (1996). Retrieved from http://www.mud.co.uk/richard/hcds.htm

Bateman, C., Lowenhaupt, R., Nacke, L.: Player Typology in Theory and Practice. DiGRA, Utrecht (2011)

Beal, C., Beck, J., Westbrook, D., Atkin, M., Cohen, P.: Intelligent modeling of the user in interactive entertainment. In: AAAI Stanford Spring Symposium. Stanford, CA (2002)

Berens, L.V.: Understanding Yourself and Others, 3rd edn. Telos Publications, Hungtington (2006)

Bedek, M.A., Seitlinger, P., Kopeinik, S., Albert, D.: Multivariate assessment of motivation and emotion in digital educational games. In: Proceedings of the 5th European Conference on Games-Based Learning, pp. 18–25 (2011)

Bethke, E.: Structuring Key Design Elements. Gamasutra (2003). Retrieved from http://www.gamasutra.com/view/feature/131281/structuring_key_design_elements.php

Björk, S., Holopainen, J.: Patterns in Game Design. Charles River Media, Hingham (2005)

Bjork, S., Holopainen, J.: Games and design patterns. In: Salen, K., Zimmerman, E. (eds.) The Game Design Reader. MIT Press, Cambridge (2006)

Brathwaite, B., Schreiber, I.: Challenges for Game Designers. Charles River Media, Boston (2009)

Breining, S., Kriegel, H.-P., Schubert, M., Zufle, A.: Action sequence mining. In: Croonenborghs, T., Driessens, K., Missura, O. (eds) Second International Workshop on Machine Learning and Data Mining in Games, at European Conference on Machine Learning. Athens, Greece (2011). Retrieved from http://www-kd.iai.uni-bonn.de/dmlg11/

Caillois, R., Barash, M.: Man, play, and games. University of Illinois Press (1961)
Chanel, G., Rebetez, C., Betrancourt, M., Pun, T.: Boredom, engagement and anxiety as indicators for adaptation to difficulty in games. In: Proceedings of the 12th International Conference on Entertainment and Media in the Ubiquitous Era. ACM, Tampere (2008). doi:10.1145/1457199.1457203
Church, D.: Formal abstract. In: Salen, K., Zimmerman, E. (eds.) The Game Design Reader: A Rules of Play Anthology. MIT Press, Cambridge (2006)
Costikyan, G.: Uncertainty in Games, 1st edn. MIT Press, Cambridge (2013)
Cousins, B.: Low-level game design, atoms, measurement and hierarchies. In: Game Developers Conference Europe. GDC, London (2005). Retrieved from http://www.bencousins.com
Cowley, B.: Player profiling and modelling in computer and video games. In: School of Computer and Information Engineering, University of Ulster, Coleraine (2009)
Cowley, B.: The QUARTIC process model for developing serious games: "Green My Place" case study. In: Lee, N. (ed.) Digital Da Vinci: Computers in the Arts and Sciences, 1st edn, pp. 143–172. Springer, New York (2014). doi:10.1007/978-1-4939-0965-0_8
Cowley, B., Charles, D., Black, M., Hickey, R.: User-system-experience model for user centered design in computer games. In: Adaptive Hypermedia and Adaptive Web-Based Systems, vol. 4018, pp. 419–424. LNCS, Dublin (2006). doi:10.1007/11768012_62
Cowley, B., Charles, D., Black, M., Hickey, R.: Toward an understanding of flow in video games. Comput. Entertain. **6**(2), 1–27 (2008)
Cowley, B., Charles, D., Black, M., Hickey, R.: Analyzing player behavior in pacman using feature-driven decision theoretic predictive modeling. In: Proceedings of the 5th International Conference on Computational Intelligence and Games, pp. 170–177. IEEE Press, Milan (2009)
Cowley, B., Moutinho, J., Bateman, C., Oliveira, A.: Learning principles and interaction design for "Green My Place": a massively multiplayer serious game. Entertain. Comput. **2**(2), 10 (2011). doi:10.1016/j.entcom.2011.01.001
Cowley, B., Charles, D., Black, M., Hickey, R.: Real-time rule-based classification of player types in computer games. User Model. User Adapt. Interact. **23**(5), 489–526 (2013). doi:10.1007/s11257-012-9126-z
Cowley, B., Bedek, M., Ribeiro, C.S., Heikura, T., Petersen, S.A.: The QUARTIC process model to support serious games development for contextualized competence based learning and assessment. In: Handbook of Research on Serious Games as Educational, Business, and Research Tools: Development and Design. IGI Global Publishers, Hershey (2012)
Cowley, B., Kosunen, I., Lankoski, P., Kivikangas, J.M., Jarvela, S., Ekman, I., Ravaja, N.: Experience assessment and design in the analysis of gameplay. Simul. Gaming **45**(1), 41–69 (2014). doi:10.1177/1046878113513936
Dalbey, J.: Pseudocode Standard. California Polytechnic website (2012). http://users.csc.calpoly.edu/jdalbey/SWE/pdl_std.html. Retrieved 1 Apr 2015
Davis, W.: Systems Analysis and Design? A Structured Approach. Addison-Wesley, Reading (1983)
Drachen, A., Canossa, A., Yannakakis, G.N.: Player modeling using self-organization in tomb raider: underworld. In: Proceedings of the 5th International Conference on Computational Intelligence and Games. IEEE Press, Milan (2009)
Galway, L., Charles, D., Black, M.: Machine learning in digital games: a survey. Artif. Intell. Rev. **29**(2), 123–161 (2009). doi:10.1007/s10462-009-9112-y
Gee, J.P.: What Video Games have to Teach Us About Learning and Literacy. Palgrave Macmillan, New York (2003)
Gmytrasiewicz, P.J., Lisetti, C.L.: Modeling users' emotions during interactive entertainment sessions. In: Proceedings of AAAI 2000 Spring Symposium Series. Artificial Intelligence and Interactive Entertainment, 20–22 March 2000. AAAI Press, Stanford (2000)
Gómez-gauchía, H., Díaz-agudo, B., González-calero, P.A.: Automatic personalization of the human computer interaction using temperaments. In: Geoff, S., Goebel, R. (eds.) Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference, FLAIRS06, pp. 352–357. AAAI Press, Melbourne Beach (2006)
Groos, K.: The Play of Animals? A Study of Animal Life and Instinct. Chapman & Hall, London (1898)
Grünvogel, S.: Formal models and game design. Games Stud. **5**(1), Online (2005)
Harley, J.M., Carter, C.K., Papaionnou, N., Bouchet, F., Landis, R.S., Azevedo, R., Karabachian, L.: Examining the Potential of Personality Traits and Trait Emotions to Create Emotionally-Adaptive Intelligent Tutoring Systems. UMUAI Special Edition (2016)

Herbert, B., Charles, D., Moore, A., Charles, T.: An Investigation of gamification typologies for enhancing learner motivation. In: 2014 International Conference on Interactive Technologies and Games, pp. 71–78. IEEE, Nottingham (2014). doi:10.1109/iTAG.2014.17

Herbrich, R., Minka, T., Graepel, T.: TrueSkill: a Bayesian skill rating system. In: Schölkopf, B., Platt, J., Hoffman, T. (eds.) Advances in Neural Information Processing Systems 19 (NIPS), pp. 569–576. Morgan Kaufmann Publishers, San Mateo (2007)

Holmgard, C., Liapis, A., Togelius, J., Yannakakis, G.N.: Evolving personas for player decision modeling. In: 2014 IEEE Conference on Computational Intelligence and Games, pp. 1–8. IEEE (2014). doi:10.1109/CIG.2014.6932911

Huizinga, J.: Homo Ludens: A Study of the Play-Element of Culture. Routledge, London (1949)

Hunicke, R.: The case for dynamic difficulty adjustment in games. In: Proceedings of the 2005 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology. ACM, Valencia (2005)

Hunicke, R., Chapman, V.: AI for dynamic difficulty adjustment in games. In: Proceedings of the Challenges in Game AI Workshop. AAAI Press, San Jose (2004). Retrieved from http://www.cs.northwestern.edu/hunicke/pubs/Hamlet.pdf

Hunicke, R., LeBlanc, M., Zubek, R.: MDA: a formal approach to game design and game research. In: Proceedings of the Challenges in Game AI Workshop. AAAI Press, Stanford University, Palo Alto (2004)

Järvinen, A.: Games Without Frontiers: Methods for Game Studies and Design. VDM Verlag, Copenhagen (2009)

Keirsey, D., Bates, M.M.: Please Understand me? Character and Temperament Types. Distributed by Prometheus Nemesis Book Co, Del Mar (1984)

Kludas, J.: Information fusion for multimedia: exploiting feature interactions for semantic feature selection and construction. PhD thesis, University of Geneva, Geneva (2011)

Koster, R.: A Theory of Fun for Game Design. Paraglyph Press, Scottsdale (2005)

Lankoski, P., Björk, S.: Theory lenses. In: Proceedings of the 15th International Academic MindTrek Conference on Envisioning Future Media Environments—MindTrek '11, p. 16. ACM Press, New York (2011). doi:10.1145/2181037.2181041

Lazzaro, N.: The four fun keys. In: Isbister, K., Schaffer, N. (eds.) Game Usability: Advancing the Player Experience, pp. 315–344. Elsevier, Burlington (2008)

LeBlanc, M.: Tools for creating dramatic game dynamics. In: Salen, K., Zimmerman, E. (eds.) The Game Design Reader: A Rules of Play Anthology, 1st edn. MIT Press, Cambridge (2006)

Ludford, P.J., Terveen, L.G.: Does an Individual's Myers-Briggs Type Indicator Preference Influence Task-Oriented Technology Use? Human–Computer Interaction (INTERACT '03): IFIP. IOS Press, Zurich (2003)

Marchiori, E.J., Serrano, Á., Blanco, Á.D., Martínez-Ortíz, I., Fernández-Manjón, B.: Integrating domain experts in educational game authoring: a case study. In: Digital Game and Intelligent Toy Enhanced Learning (DIGITEL), 2012 IEEE Fourth International Conference, pp. 72–76. IEEE (2012)

McCrae, R.R., Costa, P.T.: Reinterpreting the Myers-Briggs Type Indicator from the perspective of the five-factor model of personality. J. Personal. **57**(1), 17–40 (1989)

McCrae, R.R., John, O.P.: An introduction to the five-factor model and its applications. J. Personal. **60**(2), 175–215 (1992). doi:10.1111/j.1467-6494.1992.tb00970.x

Missura, O., Gärtner, T.: Player modeling for intelligent difficulty adjustment. In: Discovery Science, pp. 197–211. Springer, Berlin (2009)

Romano, N.C., Nunamaker, J.F.: Meeting analysis: findings from research and practice. In: Proceedings of the 34th Annual Hawaii International Conference on System Sciences, p. 13. IEEE Computer Society (2001). doi:10.1109/HICSS.2001.926253

Rouse III, R.: The Rise and Fall and Rise Again of Game Design Rules. Gamasutra (2015). http://www.gamasutra.com/blogs/RichardRouseIII/20150218/236699/The_Rise_and_Fall_and_Rise_Again_of_Game_Design_Rules.php. Retrieved 3 Apr 2015

Salen, K., Zimmerman, E.: Rules of Pay? Game Design Fundamentals, vol. 1. MIT, London (2004)

Schell, J.: The Art of Game Design? A Book of Lenses. Elsevier/Morgan Kaufmann, Amsterdam/Boston (2008)

Stewart, B.: Personality and Play Styles: A Unified Model. Gamasutra (2011). http://www.gamasutra.com/view/feature/6474/personality_and_play_styles_a_.php. Retrieved 10 Sept 2015

Walters, R.: Categories and Computer Science. Cambridge University Press, Cambridge (1991)

Wittgenstein, L.: In: Anscombe, G.E.M. (ed.) Philosophical Investigations, 3rd edn. Wiley-Blackwell, Malden (1953)

Zagal, J.P., Mateas, M., Fernández-vara, C., Hochhalter, B., Lichti, N.: Towards an ontological language for game analysis. In: Proceedings of International DiGRA Conference, pp. 3–14. DiGRA, Vancouver (2005). Retrieved from http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.105.7454

Zook, A., Riedl, M.O.: A temporal Data-Driven player model for dynamic difficulty adjustment. In: AIIDE (2012)

Benjamin Cowley BrainWork Research Centre, Finnish Institute of Occupational Health, Helsinki, 00290, Finland. Benjamin Cowley is a docent of Cognitive Science at the University of Helsinki. He obtained a Bachelors in ICT from Trinity College Dublin in 2003, and defended his Ph.D. in Computer Science at the University of Ulster, Northern Ireland, in 2009. Initial post-doctoral projects investigated psychophysiology and learning in serious games at Aalto University, Helsinki. Later at the University of Helsinki's Cognitive Brain Research Unit, he coordinated a clinical trial on neurofeedback therapy for attentional disorder. Currently working at the BrainWork Research Centre, Finnish Institute of Occupational Health, ongoing research interests include learning in people and machines, attention, and high performance cognition; using brain imaging and computational methods.

Darryl Charles Senior Lecturer, Computer Science Research Institute, Faculty of Computing and Engineering, University of Ulster, Northern Ireland. Darryl Charles graduated with a B.Eng. Electrical and Electronic Engineering from Queens University Belfast in 1988. He attained an M.Sc. in Microelectronics and Microcomputer Applications from the University of Ulster in 1995 and a Ph.D. within computational intelligence research in 1999 at the University of Paisley. He held a faculty post as a Senior Lecturer at the University of Paisley where he worked between 1996 and 2001. Dr. Charles is currently a Senior Lecturer at the University of Ulster and his academic specialism is in the areas of computational intelligence and serious games.