Generating Stories Using Role-playing Games and Simulated Human-like Conversational Dynamics

Alan Tapscott
alan.tapscott@ucm.es

Carlos León
cleon@ucm.es

Pablo Gervás
pgervas@ucm.es

Facultad de Informática
Universidad Complutense de Madrid

Abstract

Tabletop role-playing games (RPGs) have a well-tested history of making possible the improvisation of a story through the players' interactions. Adapting these human dynamics and game setting and mechanics could represent a new and fertile approach to computational story generation. In this paper we introduce a story generation system that recreates the player interaction sequence that takes place in a tabletop role-playing game (essentially a human storyteller and a player character conversation). We then process these interactions to render a story following the rules and using the knowledge base from a popular RPG. Finally, we control parameters present in the narrative RPG ruleset to tweak the resulting story, such as the presence of mental, physical and social challenges, as well as the amount of protagonism that each player has.

1 Introduction

Role-playing games (RPGs) are a plausible model for narrative. They are a successful creative exercise in which the interaction between the players facilitates the emergence of a story. Their jump into the digital media represents one of the few successful forms of interactive narrative (Tychsen, 2006). This improvised human social interaction happens naturally, albeit constrained by the rules and setting of the game. Despite the pressure of keeping the game rhythm and the chaotic nature of improvisation, the resulting stories are coherent enough for being recorded in multiple forms of media. Not only some popular books and characters have been based on real RPG game sessions that took place at some point, but lately we have how there is an audience for online videos and podcasts that record live RPGs.

Using RPGs for story management or generation is not a new idea. For instance there is work detailing the usage of a game master (GM) story facilitator (Aylett, 1999), or automating story orchestration just like a GM would (Graham et al., 2012). Other attempts include implementing a belief-desire-intent system into a computer game to perform as a GM (Luong et al., 2017) or adapting game mastering laws for interactive storytelling (Peinado and Gervas, 2004). Similarly, simulating with agents to drive the generation is not unheard-of, for instance there have been attempts that build upon conversational threads for virtual actors for interactive digital storytelling (Spierling et al., 2006) or the development and creation of characters with feelings and personalities (Curry and O'Shea, 2012). Also, some approaches explore the potential of agent-based systems, such as managing the interaction of human users with computer-controlled agents (Riedl et al., 2003), system that use autonomous believable character agents to augment a story world simulation (Riedl and Stern, 2006) a modular microservice-oriented story stage ecosystem (Concepción et al., 2018) or an implementation of agents to behave according to the scripts via automated tuning of goal parameters (Si et al., 2005). Despite the rich existing attempts to use RPGs for story generation, we believe there's still potential in them to be tapped into.

In this approach, we explore RPG-based story generation. However, instead of focusing on the rules driving the game, we propose to abstract one level and examine the interaction, the *conversation* itself as a source of narrative content. By simulating the human interactions that take place during a RPG session and implementing a game's ruleset and knowledge base derived from its setting we

Proceedings of the 3^{rd} Workshop on Computational Creativity and Natural Language Generation, pages 34–42,
Tilburg, November 2018. ©2018 Association for Computational Linguistics

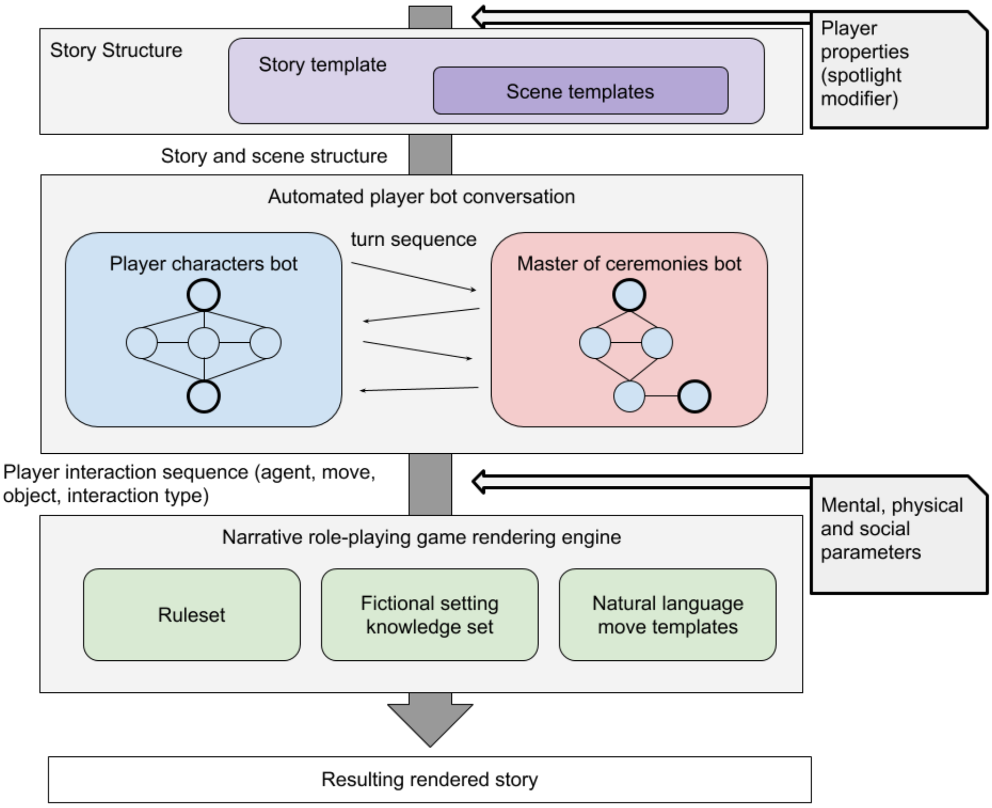
Story Structure
Story template
Scene templates
Player properties (spotlight modifier)
Story and scene structure
Automated player bot conversation
turn sequence
Player characters bot
Master of ceremonies bot
Player interaction sequence (agent, move, object, interaction type)
Mental, physical and social parameters
Narrative role-playing game rendering engine
Ruleset
Fictional setting knowledge set
Natural language move templates
Resulting rendered story

Figure 1: Story Generator structure

can generate stories similar to the ones created in a RPG session. To our best knowledge, this focus on *conversational dynamics* has not yet been exploited for creative processes in a computational system.

2 RPG-based Story Generation from Players' Conversational Dynamics

We propose to generate stories using a sequence of human-like interactions (mimicking conversation dynamics that happen in a RPG tabletop experience), a RPG ruleset (providing the base structure and dynamics) and the RPGs fictional setting knowledge base (used to fill and enrich the generated skeleton structures with the desired background and aesthetics). The artifacts provided by the generator should resemble the kind of stories the creators of the game had in mind while also being a human-like story.

The presented solution to story generation has the following key contributions:

- The conversation automation is RPG implementation-agnostic. Using distinct RPGs would produce distinct stories.
- Control over certain narrative aspects (such as character protagonism or threat types) thanks to the parameters included by narrative games.
- Streamlining of certain tasks of RPGs that typically require additional effort, such as number upkeeping, balancing magnitudes and distributing resources.

The following modules constitute the proposed story generator. Their configuration can be seen in Figure 1.

- A story structure that includes a story template containing scene templates that serves as the backbone of the story to provide context to the player interactions
- Automated player bot agents that interact with each other and produce a sequence of abstract interactions meant to resemble human conversation dynamics
- A rendering engine capable of processing abstract interactions and generate template-based natural language

2.1 RPGs: from Adventure simulation to Open Ended Narrative Improvisation

Our first concern would be to pick a RPG to base our generator on. Role-playing games are

Proceedings of the 3rd Workshop on Computational Creativity and Natural Language Generation, pages 34–42,
Tilburg, November 2018. ©2018 Association for Computational Linguistics

about playing a specific character, trying to behave as someone else in a fictional, interactive story. The popularity of their video game adaptations have obfuscated their tabletop origins and made their distinction more problematic(Hitchens and Drachen, 2009; Tychsen, 2006). The genre became popular with the Dungeons and Dragons (Gygax and Arneson, 1974) boom in the seventies. These early iterations were focused on simulating fantasy adventures, and were structurally and mechanically close to the simulation wargame hobby. One of the players had the role of dungeon master, or game master. She was responsible for introducing the setting, story and challenges to the player characters, who each played the role of an adventurer. The player characters went through the trials presented by the GM to reach the ultimate reward. From a narrative point of view, the story was almost cosmetic, and introducing the games setting's own tropes, aesthetic and ambience without much impact in the game's mechanics.

Over time, some new role-playing games introduced more flexible rulesets to put emphasis on the story. Vampire the Masquerade (Rein-Hagen, 1991) was the first game to have the golden rule, that is, to explicitly empower the Storyteller (equivalent to the GM) to modify any aspect of the game that helps in telling the story she wants to. Also, it encouraged the Storyteller to focus less in the mechanical challenge and more in the emerging story. It de-emphasized the antagonism between GM and player characters (PCs) and it encouraged their collaboration to create a story together. The resulting family of RPG games, often denominated narrative games, presents mechanics and dynamics that are meant to propitiate the emergence of a story through a human conversation. Some of the latest most proposals embrace an even more free and dynamic view of role-playing games, proposing an experience close to free-form theatrical improvisations, going as far as to removing the figure of the GM, eliminating the player asymmetry, embracing a more democratic and creative approach. Therefore, we are interested in two elements found in narrative RPGs: a discrete ruleset meant to mediate player interactions and a knowledge base with fictional setting-specific story elements and tropes.

We chose the Apocalypse World (D. Vincent Baker and Baker, 2010) (AW) ruleset to base our story generator on. Apocalypse World introduced the Powered by the Apocalypse (PbTA) engine, a ruleset that was built and functions in an innovative fashion. The PbTA game system has been significantly influential in the whole RPG industry, however we chose it for several specific reasons. The ruleset is focused on narrative improvisation, reducing the importance of pre-session story structuring and preparation. Most interactions are based on action abstractions called moves that are abstract and leave room for interpretation in meaning, causality and story impact. It also includes actual lists of elements and parameters to auto generate the story world, such as locations, objects, character names, motivations and their typical behavior. Finally, the post-apocalyptic aesthetic favors less populated and minimalist story worlds.

2.2 Input parameters and effect

We have included some input parameters that can provide control over the story. In a tabletop RPG experience, generally speaking, it is better if all player characters have a similar window of time to play their characters. The MC (AW's denomination for the GM role) is generally responsible of this. In our approach, we can limit every player character's moves, and by extension, how much presence they end up having in the final story. To do so, we introduce the *spotlight* metric. Player character bots accumulate spotlight points when either they are the agent or object of a move. Spotlight points will be then used to weight the random function used to determine what player characters are included in a scene and what player characters are either the agent or object of a move. As result, by default all player characters should be featured in a similar measure in the story. Since player bots do not care about their attention, if we desire to give more visibility to a certain player character, we can simply add a multiplier that reduces the amount of spotlight point that she earns. This metric can be used to establish a clear protagonist or to introduce secondary or even deuteragonist characters.

Our system is naive when choosing what moves to assign to what interaction beyond the action/reply/interference/end dynamics. Certain moves evoke certain story types, even within the base theme and mood of AW stories. A required element can be retrieved by either seducing/manipulating or by doing battle, however the resulting story will be quite different. To adapt this, we resort to the Mental, Physical and Social ratings (MPS) introduced in the SAS storytelling adventure system developed by White Wolf Inc (Webb and Hindmarch, 2008). SAS was meant to be a universal system for writing tabletop RPG adventures, and the MPS was used to rate narrative scenes in terms of mental physical and social challenge. The stories published using the SAS system had a global MPS rating and a scene MPS rating. Since our generated story structure also has these two levels, we quantified all the implemented moves with an MPS rating, allowing us to prejudice their incidences to achieve scenes and stories that fulfill a certain MPS criteria. This way we can either obtain stories with balanced mental, physical and social moves, or stories that favor one of the three magnitudes.

2.3 Scene and Story Templates

Our next concern would be to provide the generator with a basic structure or framework to build the story upon. While AW is focused on moves and the player conversation, it is vague in its discussion on the overall story structure. It does suggest framing the action in scenes, each involving the MC and some PCs. From a structural perspective, our system's story model is composed by scenes. Both, the story and the scenes follow a specific template that requires certain conditions to end. For the story, the implemented template is meant to represent a journey, in which the protagonists are moving to reach a specific location. For every scene, the only implemented template requires the player characters to obtain a specific element to ensure their survival. When the scene is generated, the missing element is assigned to a random threat. Once any player character does any move targeted against the threat that possesses the required element, she will automatically obtain the element and the scene will be marked as ready to finish. The nature of the move or player interaction that results in the retrieval of the missing element is not relevant for the system's functioning.

Therefore, the generated story should contain a finite sequence of scenes in which the player characters obtain elements they need to survive until they reach their final destination. The final destination and the location for every scene will be randomly generated using lists provided in the AW fictional setting. Also, the required elements, scenery elements, the present PCs and the present threats will also be randomly generated from the setting knowledge base provided by AW. In our implemented prototype, we introduced 4 PC playbooks, 2 threat types and 23 survival resources (all extracted from the AW rules) as well as 12 location names and 31 scenery elements (extracted from the AW provided examples and recommended media).

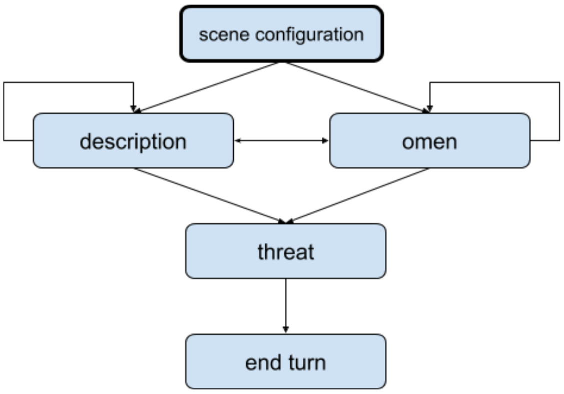

Figure 2: MC Interaction State Diagram

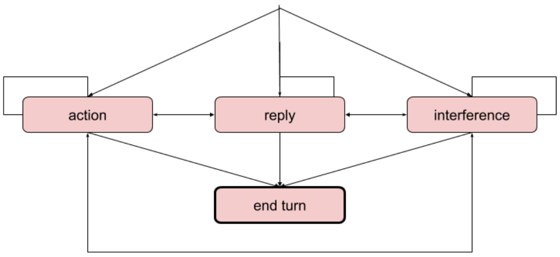

Figure 3: PCs Interaction State Diagram

2.4 Master of Ceremonies and Player Character Conversational Interactions

The next relevant module, would be the player conversation interactions simulator. In a tabletop RPG, specially in the ones that take place in narrative games, the gameplay is essentially a conversation between the players that involves certain game mechanics. For the purpose of our simulator, we need to simulate this conversation dynamics between the actual players by implementing automated bots that behave in a similar way. The resulting interaction sequence should resemble that one of a real tabletop RPG session. Note that at this stage we are not trying to synthesize the content the interactions beyond their abstract representation, this will be done at a later stage (natural language rendering).

In a tabletop RPG session there are two very distinctive events. When the MC speaks, she is behaving like a storyteller, configuring the whole fictional world, including the scenery and threats while excluding the playable characters. She sets the stage for the PCs to act or reacts to the PCs actions making the world believable and interactive. When the PCs talk, they generally start a multiple participant conversation while driving the plot forward with their actions. Our system distinguishes these two stages of the player interaction and will alternate between the MC and the PCs turn. Every turn will include one or more interactions. The MC turn, will include the following possible interactions (presented in Figure 2):

1. scene configuration: Present the context to the PCs, providing details of the location and explaining the goal of the scene. This is always the first interaction in a scene and happens once per scene.
2. description: Detail the ambience and mood of the scene. Evoke the target mood and theme.
3. threat: Use a threat (through the environment or a non player character) to provoke some sort of crisis.
4. end turn: Finish the current MC turn

On the other hand, on the player characters turn, they have the following available interactions (Presented in Figure 3):

1. action: An agent performs a general purpose move on any target.
2. reply: The target of the last move replies with another move targeting the last move's agent. The reply move can be targeted to either a PC character or a MC threat.
3. interference: Some third party agent uninvolved in the last move interrupts with another move targeting the last move's agent or object.
4. end turn: Finish the current PCs turn

The system uses a random number generator to decide the sequence of interactions that takes place in each of the alternating turns. The probabilities were assigned arbitrarily. Note that only the master of ceremonies player acts during her turn, but all player characters can potentially act during their turn. The decision to make every turn exclusive to either the MC or the PCs, not allowing to interleave each others' interactions into a hybrid turn, was based in our informal observations of real RPG games. MCs tend to drag, creating monologues to configure the scene while PCs tend to interrupt and cooperate with each other to answer the MCs interventions. This arbitrary design decision is addressed more in depth in the discussion section. Next we introduce a couple of exemplified MC and PCs turns:

The MC begins her turn with a **scene configuration** interaction, then a **description** and then a **threat** aimed at player 1. Finally she **ends** the MC turn. After the MC turn and during the PCs turn, player 1 initiates with a reflexive **action**, then adds a follow up **action** targeting player 2. Then player 2 **replies** with another action directed at player 1. Finally, player 3 steps in and **interferes**, targeting player 2 with her action. The PCs turn could **end** here. In the following MC turn, since we're still in the same scene, the MC skips the scene configuration, adds two **omen** interactions and one **threat** before **ending** her turn. In the second PCs turn, player 1 starts with an **action** interaction, followed by a **reply** to him by player 2 and an **ending** interaction. After this, the turn would switch back to the MC, and so on until the scene is over. This process would be repeated until the story template runs out of scenes.

2.5 Moves, Playbooks and Threats

In this module, we convert the previous abstract conversational interactions into actual entities that make sense in the context of the story world, moves. AW and the PbTA game system introduce the flexible concept of moves. A move is a generalization of a set of actions performed by an agent entity to an object entity, simplifying their mechanical effects while allowing a broad spectrum of expression. Some are meant to target the environment (such as read a situation), some are reactive (such as act under pressure) and some only make sense as interruptions (such as interrupting some action). The MC is responsible for working with the players to determine what moves are being performed, triggering their emergence during the unfolding interaction. The game suggests not to constraint player interaction to the available moves, but to adapt the moves to the player natural behavior. For instance, if a player wants to analyze something, she might be *reading a sitch*. If she wants to start a fight, she might be *doing battle*. PBTA games are meant to reflect heavily genre-specific tropes, so if the game was based on pulp noir novels, the player might be *investigating a scene*, or *starting a street brawl*. In the case of Apocalypse World, the whole ruleset was built to recreate apocalyptic stories such as the ones found in films of the like of Mad Max, The Road or The Stand.

In AW players are forced to pick a playbook, a character write up closely attached to the game's theme and aesthetic (e.g the driver, the exterminator or the biker). Each player may use general moves and specific moves to her playbook. All character can *read a sitch* or *go aggro*, but only the angel can perform a *healing touch*. Also, every playbook asks the player to customize certain parameters such as the character name, description or link to other characters, always offering a limited set of options. The MC does not pick a playbook, however the rules provide her with moves and threats. The moves are less specific than the PCs ones, they still outline the general principles and behavior expected from the MC player. They also encourage story beats expected from a post-apocalyptic fiction piece. Most involve depicting the fictional world, its elements and denizens, however, some involve well-defined threats. Threats are created from a limited pool contained in the game's rules such as grotesques or warlords (D. Vincent Baker and Baker, 2010, p. 138), and have specific moves associated that reflect their nature.

Our generator creates a character for every player, picking the necessary parameters from the provided set. Also, creates a pool of Threats for the MC to use. The resulting set of generated elements, will define the moves that will be possible during the simulated game session, as well as most of their potential actors and objects.

2.6 Natural Language Rendering

The last module of our system is concerned with transforming moves into natural language that makes sense to the reader. The module is agnostic of the story structure configuration and is merely concerned with the conversion from the moves into story sentences. Since Moves are abstractions that make sense in the context of a specific kind of story, they could be written in many permutations of natural language. More so, the context can influence a great deal what write ups make or not make sense for a specific move. Beyond the basic configuration of agent/object/move, many factors can potentially influence the resulting text. For the purpose of our generator, we have paired every move with an array of text templates to be filled with the domain-specific data needed for their rendering. For instance, the text templates for *read a sitch* have the following format:

{agent} observes thoroughly the {location_name}, paying special attention to the {scene.location.get_elements(random(1,3))}

Which once filled with the appropriate data, could be transformed into natural language:

Abondo observes thoroughly the cave, paying special attention to the maggots and the tubercules

The parameters get filled dynamically when using interactions with their context. The template will just replace the agent, scene location and scene location elements with the right data from the story template, scene template and interaction from the sequence. With the sequence of interactions that we generated in the previous section, our next step is to pair each interaction with a move. By pairing moves with player conversational interactions, we translate a sequence of abstract interactions into a series of abstract moves with a text representation that make sense in the context of a post-apocalyptic story. This was done in the following way for the player character moves:

1. action move: Any move that is not explicitly written as a reply or interference and requires a character object.
2. reply move: Any move, regardless of it was written as a reply or not, that does require a character object and was not written as an interference.
3. interference move: Any move, regardless of it was written as an interference or not, that does require a character object and was not written as a reply.

For the MC, this proved to be more challenging. Most of the moves enunciated were highly abstract or required complex natural language formulation we were not capable of generating in the relevant module. We introduced some base moves that followed the principles enunciated for the MC (scene configurations, descriptions and omens) while implementing the ones included in the rules that our approach was capable of (threats):

1. scene configuration: We generate a text detailing the location for the scene and what player and non player characters are present.
2. description: We generate a short sentence de-

scribing the scenery and some of its elements.

3. omen: We generate a description for a location that is not the one featured in the scene.
4. threat: These are given a more specific template, mostly involving non player characters that are hostile to the PCs.

One of the main issues with the overall approach, is that since the coupling between player interactions and moves is completely random, the resulting text might seem quite incoherent. Adding a motive of some sort is fundamental so the reader can get some notions of causality from the reading experience. For this purpose, we rely on several elements. First, the scene template (that involves recovering a missing element) provides a clear motive for the move. In this case, we would complement the result with a proper suffix: ***p1*** *does something to* ***threat1****, recovering the* ***missing element***. Another motive we use, are player history relations. All of AW's playbooks have player establish previous history between them. Our generator creates these relationships and uses them to expand the first move rendering between the relevant players: ***p1*** *manipulates* ***p2****.* ***p1*** *and* ***p2 have history*** *together*. The last element we use are the impulses of threats. AW threats not only have moves, but also an impulse. The impulse is meant to be used as the driving force for the threat to act, and is antagonistic to the players. Using the impulse, we expand the renderings of any threat's move: ***threat1*** *does something to* ***p1*** *because she has an* ***impulse***. The implemented prototype had 17 PC moves and 19 MC moves extracted from the AW rules, each with two distinct natural language templates. These templates resorted to auxiliary lists of 12 location names, 31 location elements, 100 general adjectives, 78 personality adjectives (all extracted from the AW examples and recommended media as well as an arbitrary word corpus) and 24 threat motivations (extracted from the AW rules).

3 Example

This section has a step by step execution of the story generator, illustrating the detailed process to construct a scene using our approach.

Our initial setup will involve four players. We gave all the same spotlight rating, except for p2 who will have a spotlight index of 4 to become the protagonist. MPS wise, we gave the magnitudes values of 0.25, 0 and 1. We want a social story with little physicality and a few mental interactions.

- players: p1, p2, p3 and p4
- playbooks: p1 plays Enoughto-eat the artist, p2 plays Kettle the angel, p3 plays Preen the exterminator and p4 plays III the Driver

Our story template will be the journey, which requires a destination.

Story template

- locations: The Sick Desert, the Good Mountain, the Bad Cave, the Depression
- destination: The Good Mountain

Scene template

- location: the Bad Cave
- missing resource: meat
- player characters: Enoughto-eat the artist, Kettle the angel and Preen the exterminator
- threats: Dremmer the masochist aberrant and Krin the warlord
- target: Krin has the meat

Once the templates have been filled, we can now start to generate the conversation interactions. Conversation interaction sequences are made of tuples of agent, move, object and interaction type.

We begin with the first MC turn

- `(mc, scene_conf,)‘`
- `(mc, omen, sick desert)`
- `(mc, description, bad cave)`
- `(mc, threat, p2)`
- `(mc, end,)`

Next we run the first PCs turn. Since p2 has a high spotlight index, she is more likely to be either the agent or object of most interactions.

- `(p2, reply, MC)`
- `(p2, action, MC)`
- `(p1, interference, p2)`
- `(p3, action, bad cave)`
- `(, end, )`

At this point, we would continue generating interactions until an arbitrary condition set in the template is met (for instance, after three turns for each). Social moves are more likely to be picked when converting the interactions into moves and physical move will be introduced if possible (MPS magnitudes were set to favor social and neglect physical). We will skip ahead to the rendering process.

- *Enoughto-eat, Kettle and Preen arrive to the Bad Cave searching for meat. Dremmer and Krin are dwelling there. Krin has the meat they need.*

- *Somewhere out there, there's a Sick Desert with noise and pain.*
- *The oil of the Bad Cave is common and bitter.*
- *Dremmer guides Kettle to the cave's sand.*
- *Kettle manipulates Dremmer with subtlety.*
- *Kettle uses her healing touch with Krin, obtaining the meat from him.*
- *Enoughto-eat shows some of her skin to Kettle.*
- *Preen reads the sitch in the cave, noticing the sand and the bullets.*

We have skipped several sentences, such as the rendering of the story introduction that explains the overall story objective (complete the journey), as well as the scene and story conclusion. While the results are not very specific and the rendering leaves room for interpretation, however the usage of elements from the game's setting might help readers (specially those familiar with the genre) in finding the results consistent and coherent.

4 Discussion

The presented approach relies on the emergence of sense by combining structures and knowledge from a role-playing game. This implies that the resulting stories will always be conditioned by the RPG used to convert the player interactions into moves or any other abstraction meant for the story world. This might be mitigated by combining rules and knowledge from multiple RPGs. It remains to be seen how scalable would be this approach with all the potential problems of conceptual overlapping and compromised consistency and coherence. Also, rigid text templates such as the ones we have used will quickly be spotted by any reader upon certain repetition. The rendering of moves may be improved by introducing a more complex grammar, templates parsed from related media or results enriched with predictive or machine learning approaches.

The random functions used to move between user interactions, generate names for items, traits and locations and the selection of a language template for a move follow completely arbitrary distributions based on our informal observations. As previously mentioned, we've separated the MC and PCs turn based on our intuition and informal observations arbitrarily. This is clearly a direction for future work. In order to inform the system's many choices, we might require extensive field work, analyzing real RPG sessions or even other related fictional media. The same criticism is applicable to the story and scene templates, which were designed after some of the most classic structuralist perspectives (a hero's journey that includes the acquisition of competences and the ultimate sanction).

The general goal of the generation system is to fulfill the story and scene template conditions (reach the final destination and obtain the missing elements). A common criticism to generative systems with goals is that the character and user goals are not the same. Certainly, AW states clearly that the goal is to have fun and build a story. While we acknowledge this criticism, we also argue that the failure to reach these conditions still constitutes an interesting story. Ultimately, having a simulated conversational layer (in which the goal is to entertain its participants) and a story logical layer (in which the goal is to complete character objectives) coexisting in our approach might make possible to achieve a certain balance between both objectives.

Overall, albeit the resulting text does not reflect a human quality-like result, the current text artifacts could be used to structure plots for trope-intensive genre fiction or as inspirational material.

5 Conclusions

We have described an original story generation system based on the character and game master conversation dynamics in a RPG game. We have illustrated how the implementation of the system allows tweaking player protagonism and the presence of mental/physical and social challenges. We have shown a model for shifting the focus from the pure game universe to the interactive game itself, explicitly controlling aspects such as the protagonism and the presence of mental, physical and social challenges. This enriches the generation by adding a new set of structures not present in the story world itself.

While there is plenty of room for improvement, the present approach has show how to address meta-narrative aspects and apply them into a story generation. Our present implementation constructs the final story by relying on template-based knowledge, but we believe the general model (i.e. focusing on the interaction) can be applied to other systems.

References

Ruth Aylett. 1999. Narrative in Virtual Environments - Towards Emergent Narrative. In *AAAI fall symposium on narrative intelligence*.

Eugenio Concepción, Pablo Gervás, and Méndez Gonzalo. 2018. INES: A reconstruction of the Charade storytelling system using the Afanasyev Framework. In *Proceedings of the Ninth International Conference on Computational Creativity (ICCC 2018)*.

Collette Curry and James D O'Shea. 2012. The Implementation of a Story Telling Chatbot. In *New Directions in Agent Research*. Nakanishiya Shuppan. http://nimbusvault.net/publications/koala/assr/29.html.

D. Vincent Baker and Meguey Baker. 2010. *Apocalypse World*. 2 edition.

T. C. Nicholas Graham, Quentin Bellay, Irina Schumann, and Amir Sepasi. 2012. Toward game orchestration. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction - TEI '12*. ACM Press, New York, New York, USA, page 187. https://doi.org/10.1145/2148131.2148171.

Gary Gygax and Dave Arneson. 1974. *Dungeons and dragons*.

Michael Hitchens and Anders Drachen. 2009. The Many Faces of Role-Playing Games. Technical report.

Bao Luong, John Thangarajah, and Fabio Zambetta. 2017. A BDI Game Master Agent for Computer Role-Playing Games. *Computers in Entertainment* 15(1):1–16. https://doi.org/10.1145/2629700.

Federico Peinado and Pablo Gervas. 2004. Transferring game mastering laws to interactive digital storytelling. *Lecture notes in computer science* pages 48–54.

Mark Rein-Hagen. 1991. *Vampire the Masquerade*. White Wolf, Inc.

Mark Riedl, C. J. Saretto, and R. Michael Young. 2003. Managing interaction between users and agents in a multi-agent storytelling environment. In *Proceedings of the second international joint conference on Autonomous agents and multiagent systems - AAMAS '03*. ACM Press, New York, New York, USA, page 741. https://doi.org/10.1145/860575.860694.

Mark O. Riedl and Andrew Stern. 2006. Believable Agents and Intelligent Story Adaptation for Interactive Storytelling. In *International Conference on Technologies for Interactive Digital Storytelling and Entertainment*. Springer, Berlin, Heidelberg, pages 1–12.

Mei Si, Stacy C. Marsella, and David V. Pynadath. 2005. Thespian. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems - AAMAS '05*. ACM Press, New York, New York, USA, page 21. https://doi.org/10.1145/1082473.1082477.

Ulrike Spierling, Sebastian A. Weiß, and Wolfgang Müller. 2006. Towards Accessible Authoring Tools for Interactive Storytelling. Springer, Berlin, Heidelberg, pages 169–180.

Anders Tychsen. 2006. Role Playing Games-Comparative Analysis Across Two Media Platforms. Technical report.

Eddy Webb and Will Hindmarch. 2008. Storytelling adventure system. https://www.drivethrurpg.com/product/54404/Storytelling-Adventure-System-Guide.