Asynchronous Dialogue System

Alfred Andersson, Keman Nguyen

Computer science
Bachelor's Thesis
15 ECTS
Spring 2022
Supervisor: Jeanette Eriksson
Examinator: José Maria Font Fernandez

Asynchronous Dialogue System

Alfred Andersson, Keman Nguyen
Faculty of Technology and Society
Malmö University
Malmö, Sweden
alfred.a.064@gmail.com
keman885@gmail.com

***Abstract*—Conversations between the PC (player character) and NPCs (non-player characters) in conventional games are usually sequence-based. The NPC talks to a certain point before pending the player's input, sometimes consisting of several prepared actions displayed on the screen in order to advance the conversation. While this method does provide the ability to converse within video games, our study shows it lacks the immersiveness that asynchronously based dialogue provides in some scenarios. Interruptions occur in real-life conversations and may add to a more convincing interaction. In this paper, we present a novel dialogue system that incorporates interruptions alongside emotion, making it possible for different participants involved in the conversation to interrupt and speak over each other while also having lasting consequences. This approach improves conversational players' experience by increasing character believability and engagement. For illustration purposes, interruption was integrated into a text-based game encompassing two variations of the same scenario. The study involved playing both variations of the same game, one being a traditional sequence-based conversation while the other had a fluent dialogue which supports interruption both from the PC and NPCs. Eight students previously familiar with video game dialogues played both variations, half starting with the other version. Each test ended with a survey followed by an interview talking about the answers. Each test took 30-40 min.**

***Keywords*—Video Game, Dialogue System, Dialogue Graph, Game Dialogue, Unity, Interruption, Emotional Values**

I. INTRODUCTION

The strive for immersive experiences in video games has been desired for a long time. While graphical advancements have been made, conversations between the PC (player character) and NPC (non-player character) have not seen the same evolution.

Numerous research already exists [1], [2], [3], [4], [5] with the desire to enhance the immersiveness through improvements to conversations in video games, but there are still advancements to be made.

The use of verbal and non-verbal communication between actors is one way to convey intent to one another [6]. Similarly to real life, we use communicative methods to express our intentions to other participants—if that is to the PC, NPC, or even NPC to NPC. Game designers use it to submerge the player into the world they created. In a quest to find evidence to oust the village charlatan from the village, the intent may be presented verbally to you by convincing you that the NPC's and PC's intentions align. Domsch [7] describes how those instructions can both be ludic — directed at the player — and diegetic — happen within the fictional world — at the same time. Ludic because it presents a scenario of actions as a rule that the player needs to decide between; diegetic because it is related to a narrative dilemma in which the PC's actions you choose have consequences towards the side-quest and its participants. “While ludic communication has the function of giving the player feedback about her actions and of informing her about game rules, diegetic dialogue is all about the representation of a convincing and immersive fictional world in which the player’s actions have significance” [7]. This is relevant knowledge because it encapsulates the importance of a narrative

thread, blending the ludic and diegetic together to immerse the player into the world and make them believe their actions have an impact. The designers' intention is to create a scenario where the player can engage with its mechanics and simultaneously follow its narrative path; this is accomplishable with game dialogue. Dialogue, while powerful and highly used in games — limits the impact and options that the player has both ludically and diegetically. The player is limited to a finite number of available actions to choose from. What the player wants and what the game provides may not be the same and consequently affects the narratively impact that the player has over the world. The consequences are not what is significant, but rather the illusion that the player has the power to defy the playing field. A common way is to represent a dialogue as a traversable tree where each junction serves as a choice or action that the player can take [8].

Games such as the latest *Horizon* game — *Horizont: Forbidden West*[1]— and the original *Mass Effect*[2] are dialogue-driven games taking advantage of the tree approach. The commonality of this approach is the sequenced nature of the conversation. The dialogue can be interpreted as turns that alternate between the participants. On occasion, the player is presented with multiple actions that the PC can say or do. Those choices constitute a turn for the PC before transferring the discourse to another conversational participant. The conversation is always limited to one participant at a time. As a consequence, dialogue in video games may be perceived as artificial because of the finite structure of the conversation.

Brusk [8] describes how we usually take turns in dyadic communications to prevent overlapping speech in the so-called *transition relevance place* [9] — meaning, systematic turn-taking between participants in a conversation. Turn-based conversations have been the standard convention to drive conventional dialogue in games; the issue is that interruptions still occur in real life. Interruptions pose a challenge when implementing a dialogue system. The usual sequence-based approach of one participant at a time is no longer applicable. "Interruptions, by definition, do not fit; consequently their treatment has implications for the treatment of the normal flow of discourse" [6] — meaning to pursue interruption, the way to interpret a discourse has to change.

We wish to contribute with an alternative solution capable of immersive dialogue between the PC and NPCs in games through the implementation of interruption (see section 2.A) and emotion (see section 2.B). This paper will therefore focus on a novel video game dialogue system, which uses interruption with varying repercussions depending on the timing of the interruption and emotion (see section 2.C). The timing of interrupting the other participant by selecting an action and the particular dialogue alternative the player chooses will both have an impact on the conversation. The dialogue system will simulate more realistic conversations where the content of the dialogue successively reveals itself based on time. During a conversation, different options to choose from will appear and disappear based on how relevant they are to the current state of the discourse. Picking an option will interrupt the other participant and modify the emotional values of that actor, resulting in possibly changing the relationship and topic of the conversation. The specific choices and timing when the player chooses to interrupt will affect the relationship with the other participants and the current subject of discussion. Interrupting the NPC using cooperative interruption would for example have a positive impact, while competitive interruption would have the opposite effect. The relationship values will determine how the other participants respond and interact with the player.

The paper will answer how the ability to interrupt mid-sentence while conversing with an NPC changes the conversational experience for the players compared to the traditional turn-based interactions in games and how to make the players feel like their actions during conversations have long-lasting consequences by using an Asynchronous Dialogue system (more in section 2.C).

[1] Horizon: Forbidden West. Guerilla Games (2022).
[2] Mass Effect. BioWare, Edge of Reality, Demiurge Studios, Straight Right (2007).

II. BACKGROUND

This section introduces previous research related to this study and describes concepts regarding interruption and emotion within video games which are necessary to understand the artefact.

A. Interruption

Interruption in spontaneous discourse — the most common type of dialogue when creating an immersive conversation — can generally be categorised into two distinct types: competitive vs. cooperative [10]. Li-chiung [11] describes competitive interruption as taking higher priority over the current conversational participant's speech when they intend to continue, while cooperative interruption occurs when supporting or reinforcing the current conversational participant's point without intentionally disrupting their continuation. For example, at a crime scene, the player role-plays as police who are instructed to interrogate a witness whom the player knows is hiding something from them. The witness has the personal interest to dictate the flow of conversation over you. They will try to deflect subjects given to them. It is up to the player to interrupt the witness and point out the information extracted so far during the interrogation before moving to the next subject. This interruption would be considered competitive. If you interrupt too soon, you will miss out on important information and might be perceived as too aggressive. This mechanic would add a gameplay element to the dialogue where you might need to make rapid impulsive responses during a duration or miss out on crucial actions. In another scenario, same setting — the player is involved in a dispute between a husband and wife arguing over the murder of their daughter. Both are suspects, and currently, the wife has the leverage in the conflict. You can decide to defend the husband when he struggles for a comeback. He might be more willing to help you as a consequence while lowering the chance of getting any information from the wife. If you do not intervene and instead wait for them to be done, he might be too upset to be of any use. This interruption would be considered cooperative. Both the scenarios can benefit from using interruption as both a mechanic and narrative device to enhance the experience. It is not the added choices which are important, but rather the belief that you, the player, outsmarted the game.

Dialogue in video games that implements interruption needs to take multiple conversational participants into consideration. For a game dialogue, this can be interpreted in two different ways. We will call them mechanical interruptions and narrative interruptions.

Mechanical interruption is disruptive actions the PC can do while another conversational participant is speaking. It affects the current state of the game and requires the developer to consider the outcome of that action. One possible implementation is having the current state changed to the PC. This change would immediately take into effect — or possibly pending the interruption to a predetermined gap in the dialogue before being executed.

Mechanical interruption correlates with ludic and diegetic instructions [7] where you, the player, in ludic, are presented with a rule in which different actions are given to the player to choose from. The player can decline to intervene at all if so wished. These actions may only exist during a specified duration and notify before disappearing. Similarly, with diegetic, the choices will have consequences that the player needs to take into consideration before deciding.

Narrative interruption is a scripted scenario between conversational participants involving talking over each other. This outcome will always happen in that scenario but requires the dialogue system to support multiple participants speaking — if that is in written or spoken form; this means participants' speech can overlap. In written form, one solution could be to display the second participant on top of the screen while spoken allows for multiple audio files to play simultaneously. The previously mentioned example with the player breaking up the arguing between the husband and wife is a mechanical interruption because it affects how the dialogue will flow. If the husband and wife talk over each other, that would be a narrative interruption because it does not change how the conversation will unfold — only the player can change that.

Implementing narrative interruption does not prevent using a tree-based dialogue structure, only that nodes should not interpret as finite states, with only one

being the current one. Ideally, implementing a dialogue tree supporting narrative interruption should have at least two nodes being the current states or at least the possibility to overlap the nodes over each other would allow for asynchronous conversation (more in section 6). Mechanical interruption can take advantage of narrative interruption by having the PC speak over the current conversation resulting in an overlap over the other participant to break the turn-based structure of the discourse that only allows one conversational participant to speak at a time.

Mechanic interruption does not in itself need any narrative interruption. It is possible to implement mechanical interruption to an already existing finite-state machine. In a traditional RPG (role-playing game), a turn corresponds to one dialogue paragraph with the possibility to go to the next paragraph. It could require each time input from the player but could also be time-based and moved to the next one automatically. On occasion, the player is given a choice of actions to pick from. Instead of having all the options being dependent on a player's turn, the game will instead successively present possible choices that the player can do or say for each turn that passes with the possibility of abstaining from intervening. If, for example, an NPC is describing something extensive — you, the player — always have the possibility to give an excuse as to why you are no longer interested in listening to the explanation. If a turn requires input from the player, in that case, having a choice being "..." could indicate as not interrupting.

B. Emotions

Interruptions are only one potential solution to improve the perceived immersion in a game. Numerous research on how to improve dialogues already exists with the sole purpose to enhance the perceived immersion of a discourse. One possible way to increase player engagement is to have the player interact with believable characters, with the sense of making the characters feel alive by them expressing emotion [12]. Previous research [1] has shown that players find interactions with NPCs more engaging and immersive if their emotions can be manipulated. By making the player input from the interactions invoke emotions, it is possible to create the illusion that the NPCs have personalities and the ability to adapt and react as if they were alive. Additionally, it makes the choices of the player more meaningful. For instance, if an interaction between two actors has been going poorly, they most likely would not interact the same way as they did when they had their first interaction. However, the current standard for conversational systems does not consider the impact of previous conversations and the emotions of the NPCs.

Considering the problems previously mentioned — where communication between the PC and the NPC unfolds — lacks a perceived illusion of consequences.

C. Combined

In the game Façade[3], you play as Diana, who reunites with her old college friends — Grace and Trip, a married couple — with whom you were responsible for hooking up. The player is invited to their apartment. Their marriage is not going too well, and the player gets dragged into their arguments and psychological mind games. The main story gets played out through interacting and chatting with Grace and Trip. It is possible to interrupt the other participants (Grace and Trip) mid-sentence during the conversations; this causes no lasting consequences other than being silenced or the discourse moving to the next subject. The discourse is incapable of backtracking, and the illusion of fluent conversation is under the hood turn-based. The PC can only respond to the current subject of the discourse. In our implementation, interruptions would affect based on the context, long-lasting consequences — negative or positive. By the use of emotional values, the game can track how the PC's interactions with an NPC affect its emotional feeling toward one. With interruptions: interrupting an NPC while uttering important information would be competitive and perceived as rude and negatively affect the friendliness value, or a distinct value — monitoring a patient value that works as a buffer before influencing the friendliness values. In this example, the NPC has tolerance against interruptions, but if you interrupt too frequently, this causes prolonged consequences. In another scenario: if an NPC struggles to utter their intention — interrupting and assisting with

[3] Façade. Procedural Arts (2005).

delivering the sentence would be cooperative and perceived as advantageous. When Trip attempts to defend himself in an argument with Grace, the PC can choose to support either party or none. In one scenario, the PC attests that Trip's idea sounds great when he struggles to come up with a counterargument. In a hypothetical version, meaning if the game implements interruption and corresponding consequences, the consequence would be that Trip is satisfied while Grace gets upset; their emotions would be monitored by emotional values that store all necessary data to decide how the flow of the discourse should traverse. In another scenario, the PC continuously interrupts with unsatisfied utterances — this, in turn, makes both Tris and Grace impatient and asks Diana to leave.

III. MOTIVATION

The goal of this study is to explore ways to enhance the player experience during dialogue. Using the knowledge of previous works related to emotion [1], [2], [3], [4], and interruption to possibly improve or give an alternative to current solutions, might result in more impactful and memorable video game dialogue.

IV. RESEARCH QUESTIONS

The objective of this research is to answer the following questions:

RQ1: How does the ability to interrupt mid-sentence while conversing with an NPC change the players' conversational experience compared to the traditional turn-based interactions in games?

RQ2: How can we make the players feel like their actions during conversations have long-lasting consequences by using our Asynchronous Dialogue system?

V. METHOD

The methodology used for this paper is the design research methodology by Peffers et al. [13] — which is suitable for this paper because it includes the design, development and evaluation of a new and novel artefact.

The design research methodology is an iterative process consisting of six activities in a nominal sequence. The method iterates back from activity 5 to 2 depending on the demand, meaning that it iterates back if there is any necessary missing data required for the evaluation in order to optimize the artefact and test used in the user study.

A. Problem identification and motivation

There is a considerable quantity of research in the pursuit of immersive-driven interactions with NPCs in games and more specific verbal communication. Some focus on the automatic generation of dialogue [2], [5], by making the NPC's replies feel natural, while others focus on adding emotion, relationship and familiarity [1], [2], [3], [4], to make dialogue more immersive. However, none of them has been taking the impact of interruptions into consideration when trying to mimic real-life conversations. Interruptions are inevitable and a prerequisite for eloquent utterance in discourse between two or multiple participants [6]. Instead of limiting the dialogue to one conversational participant at a time and having the game decide when you should be able to utter, instead, the PC is allowed to intervene while another participant is speaking.

Finding a solution to address the neglected issue is desired to further expand immersive interaction in dialogues, taking emotions and relationships into consideration, meaning an iterative extension to an already existing solution. By combining the solutions — discovering another way to implement a dialogue system may advocate further interest and research in interactive dialogues in video games. By laying out a framework, the reader should be able to replicate the fundamental concept of this artefact.

To scale such a system requires proper tools and editors to handle complex dialogue trees. A dialogue system requires an environment appropriate for both the player and developer; such a system lacks current research and clarity — something this paper desires to address.

B. Define the objectives for a solution

The objective is to create a dialogue system as the artefact that implements solutions for the aforementioned issues. As previously mentioned — combining the result from other implementations and support for interruptions is the main objective. The second objective is to evaluate if there is any experience difference between dialogue with and without interruptions. By examining the difference, understanding if such a mechanic can add new ways to experience a game — is feasible. The experiment is conducted by creating a game with two play scenarios — with one implementing said interruption. A Further explanation of the test can be found in section 5.D.

The purpose of the artefact is to create an immersive conversation between the PC and NPC. In this scenario, immersive refers to the ability to interrupt the other participants and conversation decisions resulting in long-lasting consequences when it comes to relationships. The developed artefact has the following additional objectives:

1. Implementing the possibility for interruptions while a conversational participant (NPC) is uttering. While another participant other than the PC is the current conversational participant, the PC can choose to interrupt mid-sentence. The PC can also choose to abstain from interrupting and wait for its turn. PC turns still exist where the other participants wait for the PC to answer; this is time-based and works similar to other interruptions. The NPCs will perform small-talk whilst waiting on player input. If the PC abstains from all choices, the conversation will continue to the next subject. Abstaining is another type of action that will result in taking a path.
   Traditional PC turns are still feasible where the other participants will wait indefinitely.
   Interruption is still a feature that the developer explicitly needs to implement and is identical to other actions — constrained to the rules set by the developers.

2. Having different dialogue actions appear and disappear during a conversation. The PC's action might only be relevant for the specific subject and fade away when engaging with a later discussion topic. It would be odd to bring up something that is currently not relevant, but this could also be something that is intentional.

3. Making the PC's choice of action influence the relationship between the PC and the specific NPC involved in the conversation. Storing emotional values that can be affected by the actions can influence what the PC can say or do and how the NPC will behave towards the PC.

4. Making the PC interruption's timing influence the relationship between the PC and the specific NPC involved in the conversation. Allowing the conventional participant to speak their intent or interrupt before complete utterance may cause consequences.

5. Interface — allowing the developer to create new dialogues from an editor that are expandable with custom nodes to express further their ideas.

C. Design and Development

The artefact is developed using the Unity Game Engine (2021.3.2f1) and taking advantage of their graph view interface. Graph View is a framework that allows the developer to create graphs consisting of nodes connectable to each other and more. Section 6 provides a more elaborate description of the creation process and the purpose behind the design choice, but a summarization describing the different modules will be given:

The dialogue system can be divided into two distinct components: the editor for creating dialogue graphs, and the runtime that can utilise a created dialogue graph. The dialogue graph editor is structured using different types of nodes. The currently implemented node types are *Dialogue Node*, *Condition Node, Basic Node, Property Node, Action Node, Blend Node, Set Node, Input Node* and *Marker Node*. By using

these nodes and connecting them with each other the designer can create dialogues.

To test and evaluate our contribution, we created an artefact that incorporates the necessary features, accessible in an appropriately tailored game environment. The game environment is a conversation scenario between the PC and an NPC. Since the focus of the game was the dialogue, we made it as minimalistic as possible, only containing the dialogue text and the sprite avatars representing the NPCs. The player is conversing with buttons representing the PC's actions. Our artefact combines most of the previously mentioned implementations into a functional interactive dialogue. Because of the game length, emotions were removed in the second edition of the game, the reason being that the player would be unaware of its effect on such a short game. Only the interruption addition will be tested; this paper assumes the results from previous research around emotion in conversation. The artefact's purpose is not meant to be a full-fledged implementation of a complete dialogue system but rather a proof of concept that demonstrates its potentiality for future research.

D. Demonstration

The demonstration follows Sali S et al.'s [14] approach using different versions of the same game — with each being a different implementation of the questions asked. Some of the questions used are inspired and repurposed by their questions.

In order to assess the impact of interruption in text-driven video games, we evaluated the presented system by letting the players play a game containing two different versions (CLASSIC and INTERRUPTION) of the same scenario, meaning that both versions of the game had exactly the same dialogue routes and amount of choices. The only difference between the different versions of the game was that one gave the pc and NPCs the ability to interrupt live while the other used the traditional turn-based dialogue system. When performing the evaluation, the players were only allowed to play each version once.

We made half of the participants start on INTERRUPTION while the other half began the CLASSIC to ensure that the order of the versions did not affect the results. The evaluation was conducted with the help of eight participants with similar familiarity with video games containing dialogue. The participants were given a short explanation of how the game worked and its backstory before playing the two versions and were then asked to fill in a form after they had completed the experiment. The participants were instructed to freely interact with the two available NPCs by choosing the different options presented to them as buttons appearing on the screen during the game. The NPCs interact with the player through text, and once the player finishes the conversation, the scenario ends.

After the participants had completed both versions of the same scenarios, they were asked to answer a questionnaire regarding their experience with the different systems. Some of the questions are: "*How engaging did you find version CLASSIC? How much influence did you feel you had over the story in the different versions?"* were answered through a 5-point scale, while the more general questions such as: "*Which one was the easiest to use and why? Did it feel like the decisions you made during version INTERRUPTION had lasting consequences which affected the interactions with the NPCs? Motivate the reason for the given answer."* were answered through free text. The questionnaire was combined with a verbal interview with the participants to clarify the motivation behind their answers. The questions in the written questionnaire are:

Q1: How engaged were you in the different versions of the game? Can you pick your favourite?

Q2: How much influence did you feel over the story with the different versions? Whom would you say has the most choices?

Q3: Which one was the easiest to use and why?

Q4: How did you form strategies and make decisions? Did those strategies work out?

Q5: Did it feel like the decisions you made during version INTERRUPTION had lasting consequences

which affected the interactions with the NPCs? Motivate the reason for the given answer.

E. Evaluation

Data from the demonstration phase is evaluated by comparing the results of the two versions of the artefact. The traditional dialogue will be compared to our dialogue additions through the feedback from its intended audience; This is done with the main purpose of evaluating the implementation of interruption and if it results in more engaging and immersive dialogues within video games.

F. Communication

The communication of the problem and its importance, together with an artefact, providing a solution is accomplished through this paper. Furthermore, a discussion evaluating the results from the tests will also be provided.

VI. IMPLEMENTATION

The artefact consists of different node types that are used in the dialogue editor and later utilised in runtime to produce an interactable dialogue with an NPC. The node types used in the dialogue editor are the following nodes: *Dialogue Node*, *Property Node, Basic Node, Condition Node, Action Node, Blend Node, Set Node, Input Node* and *Marker Node.*

- The *Dialogue Node* contains the NPC dialogue (text), an input connection and the different actions that the player can pick. Conditions are attached to the player's choices to unlock or lock the actions depending on the relationship (emotional values) the player has with the given NPC. The whole dialogue of an NPC is split into multiple *Dialogue Nodes*, where each node acts as a point in the conversation where new dialogue choices (actions) for the player appear or disappear. After an action has been selected, it will transition from the current node to the next *Dialogue Node* that the selected action is connected to. Not picking any action at all after all the NPC dialogue has been displayed is also considered an action.

- The *Property Node* uses the blackboard to hold the emotional value of an NPC. It is updated in real-time and used to track the relationship between the NPC and the PC.

- The *Basic Node* holds a value that currently can be an integer, float or bool, mainly used for *Condition Nodes*.

- The *Condition Node* makes it possible to compare a *Property Node* with a *Basic Node* — if the chosen statement (equal, greater, greater or equal, less, less or equal, not equal) is true, then the output of the node will be true, else it will be false. This node can be connected to a player's action to make it be able to lock and unlock depending on the *Condition Node* output.

- The *Action Node* contains an action that the player can choose similarly to the *Action Node* that the *Dialogue Node* locally contains. The difference is that the action from the *Action Node* can be used by multiple *Dialogue Nodes*.

- The *Blend Node* starts another *Dialogue Node* before the current one has ended, resulting in parallel dialogues during the transition between *Dialogue Nodes*.

- The *Set Node* is used during the transitions between nodes to modify blackboard properties.

- The *Input Node* is the entry point for a dialogue. The node is used to connect the runtime code to the dialogue graph, resulting in runtime events having the ability to trigger conversations.

- The *Marker Node* is used to access external code with the purpose of activating different events.

The emotion and relationships are implemented through the following methods:

- Each NPC stores emotional values such as familiarity, respect, anger etc.
- Each player action involved with an NPC will slightly modify the NPCs' emotional values.
- The emotional values are used in the *Condition Nodes,* which are connected to player actions. When the statement in the condition node is true, it will unlock the action it is connected to, if not, the action will remain unavailable for the player.

A. Artefact

The artefact is divided into three different segments — each explaining and discussing the how, why and possible alternatives related to the work. These are:

- Game: Both variations used for the test.
- Editor: The dialogue graph editor used for creating new dialogues.
- Runtime: The dialogue system that reads already created dialogue.

Because of scope limitations and time for this artefact, not all features can or are fully implemented, which a complete dialogue system may have. The game that will incorporate this system does not take advantage of all the features because many of them are targeted to more complex games. These constraints are:

- Only text-based. The game alternatives will be purely text-driven because of the limited scope. Ideally, interruptions enhance immersion in a voiced conversation.
- Limited to two conversational participants at the same time. The use of more than two conversational participants at once in a text-based game would be too confusing to follow.
- Only support for *abstract response* [14] — meaning actions rather than sentence interface. It currently only allows for short descriptions for each action the player takes.
- Not emotions. The game duration is too short for any meaningful effect on the relation between the NPCs and PC. Emotions are supported but would not create any useful results from the test that specific test interruption would only minimise the already existing choices that the player can take.

B. Game

The game is visually inspired by the game Undertale[4]. In this game, the player has been invited to a tea party high up and far away from civilization. The host — Lady Madam (seen in Fig. 1) — is a candidate to become the future monarch of the kingdom with the desire to convince the player to support her during the coming royal election. On her right side stands Charles the Bug Butler (seen in Fig. 2) — who has been her loyal butler for many years and many more to come. Their relationship is complex or right-out abusive for Charles; this does not mean Lady Madam does not care about Charles — she is just imprudent.

The game is a visual novel where the goal is to experience the narrative based on the players' decisions. There are multiple endings which span different lengths depending on the choices made. There are no right or wrong choices — only consequences depending on the players' actions.

The game contains two different versions of the same scenario; one has interruption and another with a more traditional dialogue flow. In the INTERRUPTION version, various player actions will appear and disappear during the conversation depending on how relevant they are to the current state of the conversation. You, the player, can either interrupt the NPC or let them speak

[4] Undertale. Toby Fox (2015).

and in the same way, NPC can both interrupt each other but also the PC. Different scenarios will play out depending on how you decide to lead the conversation.

This game follows *abstract response* [14] — which means short actions that do not give the full context nor what the PC will say exactly. The game plays in real-time, and the dialogue moves to the next one without any input from the player in the INTERRUPTION variation. The only time it does wait for the player is when there is a choice to pick in the CLASSIC version. It is possible to play through the whole game in the INTERRUPTION version without any player input. It is achievable to simulate silence in the CLASSIC version — with the catch being the player is required to actively press "..." each time a choice appears to abstain from partaking.

When an NPC or sometimes the PC interrupts, it will display the text on top of the screen — otherwise, it will show at the bottom of the screen. Multiple endings exist, and there are different paths under the game that the player can take. Some actions in the INTERRUPTION version require quick decisions, or else there may not be another opportunity.

Because of the short demo length, the emotional values were cut out of the game because it limited the number of choices that the player could choose depending on what actions the player had made. The study conducted was solely on the effect of interruptions in a text-based game and did not affect the study's purpose.

Fig. 1 Lady Madam greeting the player.

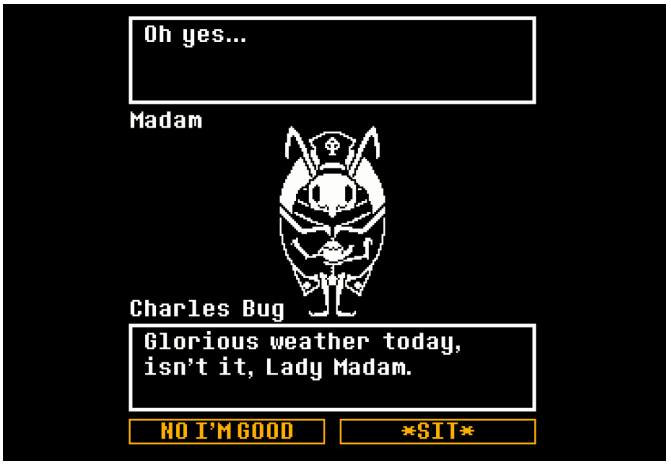

Fig. 2 Charles the Bug Butler is interrupted by Lady Madam while multiple choices are on display.

C. Editor

The dialogue editor has one purpose: the creation of node graphs saveable to files that are readable at runtime. The editor generates a file that contains all the necessary information to create a usable node structure to be evaluated at runtime. The dialogue is represented by a node graph where different types of both text and logic nodes are used to create a fully functional dialogue that includes logical behaviour such as emotional values.

The graph editor is built on top of Unity GraphView, which is generally used to visually display, edit and connect different types of nodes with each other.Some inbuilt systems within Unity that also use Graphview are Shader Graph used to create different shaders and Visual Effect Graph used for GPU particles. Our graph editor – called the dialogue editor – allows the developer to place different types of nodes and connect them to each other depending on the port type in order to create dialogue. All nodes have either input ports, output ports or both. The input ports are on the left side, while the output ports are located on the right side of the node.

The nodes can be selected, moved, copied, pasted and deleted. Different node types are accessible from a search window (see Fig. 3) categorised into groups. For example, the *Basic Node* can contain all primary value types used to hardcode certain behaviours.

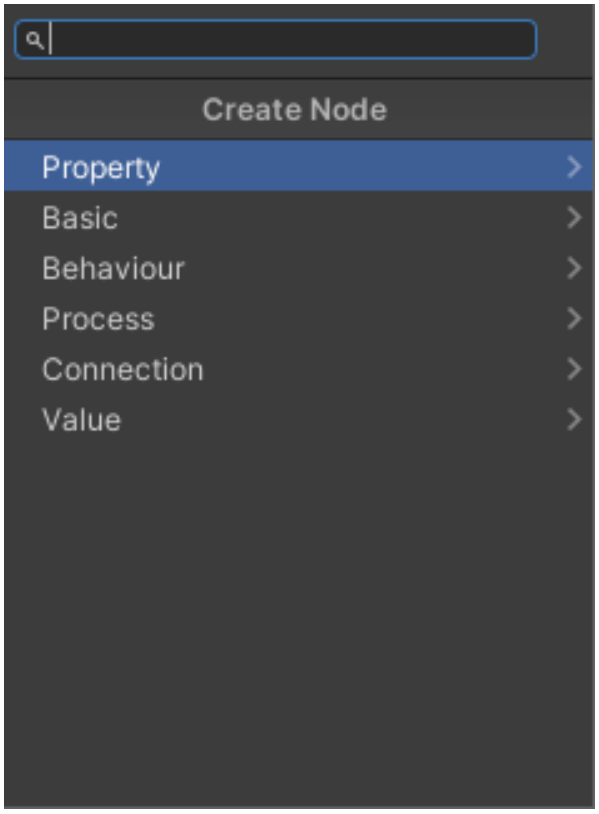

Fig. 3 Node search window

In Fig. 4, the graph consists of *Dialogue Node*, *Condition Node*, *Property Node*, *Basic Node*, *Action Node*, *Emotion Node*, *Blend Node* and *Marker Node*. Fig. 4 does not contain The *Set Node* but is instead shown in Fig. 6. Each node has its own purpose used to create a game scenario; in this case, the conversational participants are between two NPCs and the PC. An example of how these nodes can be used to create a short conversation is shown in Fig. 4.

The *Input Node* (point 1 in Fig. 4) works as the starting point of the dialogue, which determines which *Dialogue Node* will show its content first. In the example, the *Dialogue Node* at point 2 in Fig. 4 will be shown first since it is connected to the *Input Node*.

The *Dialogue Node* contains text and has the properties *Speed* and *End Offset*. These properties affect how fast the text will be revealed and how long the node will wait after all the text has been revealed before it transitions to the next node connected to the output port named "Continue". The *Dialogue Node* can contain multiple actions, each representing things that the PC is capable of doing. Multiple actions can be available simultaneously, each representing a button choice for the player. An action may only exist for a time window before disappearing. Not selecting any action will result in a transition to the node connected to the output port "Continue" while selecting one will interrupt the other actor in the instance it is selected, causing the rest of the text from the current node to not be revealed. If there is not a node connected to the output port "Continue" the conversation won't progress unless the PC picks an available action. It is possible to lock and unlock actions through the use of the *Condition Node.*

Condition Node is a type of logic node that compares two different values with one another. As shown in point 3 in Fig. 4, there are different types of comparisons to choose from that will return a condition, meaning — true or false. The input is of a numeric type — that derives from a primitive (*Basic Node*) or an external property (*Property Node*). The different actions are revealed to the player when input on the condition port is true. When there is no *Condition Node* attached to an action, the value is true by default.

In Fig. 4, there are two different actions connected to the first *Dialogue Node* named response 1 and response 2. The action labelled as response 1 will only appear to the player if the *Property Node* "NPC friendliness" used as an emotional value is greater or equal to 5. If response 1 is chosen the dialogue above (points 4 and 5 in Fig. 4) will be played out and the dialogue will end on the *Marker Node* (point 6 in Fig. 4), which works as an exit point that is capable of connecting to other parts of the game, mainly used to identify how the trigger events in runtime. In Fig. 4 it is used to activate an event to exit the game. For that reason, each of the possible outcomes (points 6, 13 and 10) of the dialogue is therefore linked to a unique *Marker Node*.

The Second action connected to the first *Dialogue Node* is through the use of the *Action Node* (point 7 in Fig. 4). The difference between the actions created through the *Dialogue Node* and actions created through the *Action Node* is that the actions that are created in the *Dialogue Node* can only be used within the given node, while the actions from the *Action Node* can be used in multiple *Dialogue Nodes*. In the example in Fig. 4, response 2 is connected to two *Dialogue Nodes* causing the action to be available during both the connected

Dialogue Nodes (points 2 and 11 in Fig. 4). Selecting an action will interrupt the other actor as previously mentioned but through the *Blend Node* (point 8 in Fig. 4) the interruption can overlap. Instead of the other actor becoming silent after the interruption has started, the player and the NPC will talk over each other. The duration of the overlap is set through the value input.

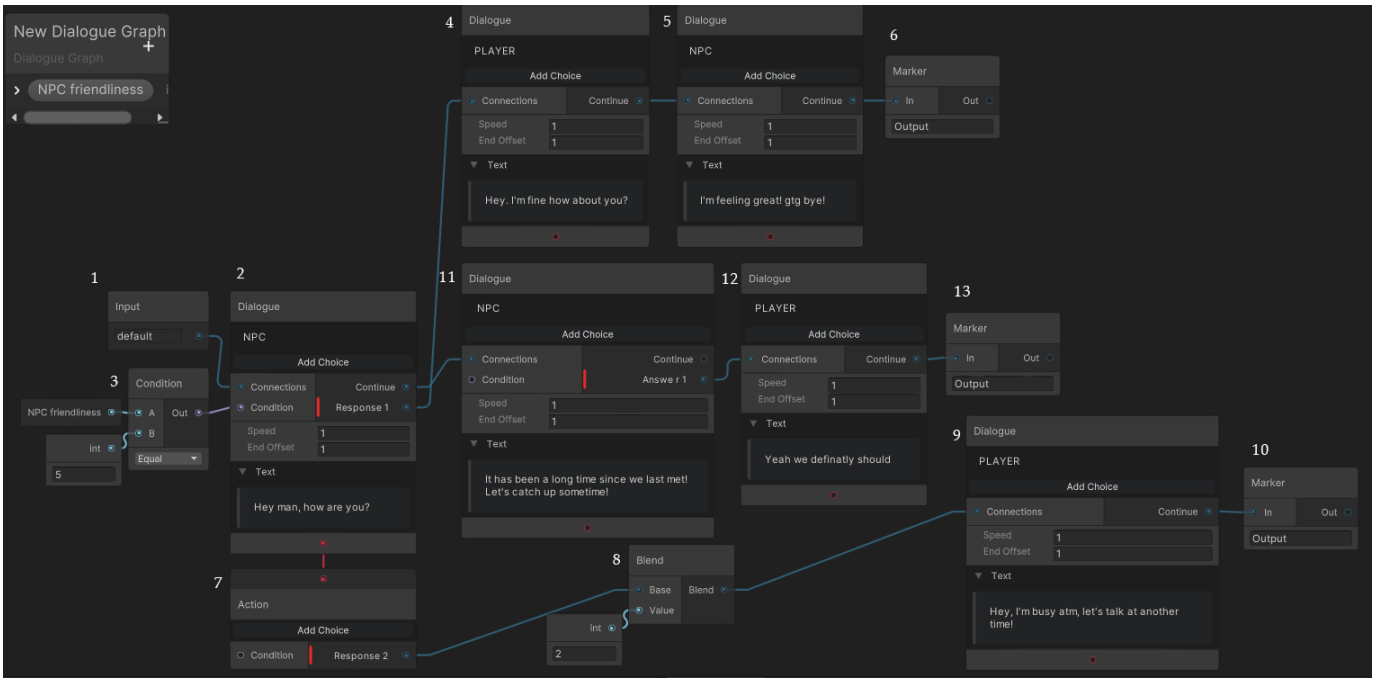

Fig. 4 Test scenario using all Nodes except Set Node

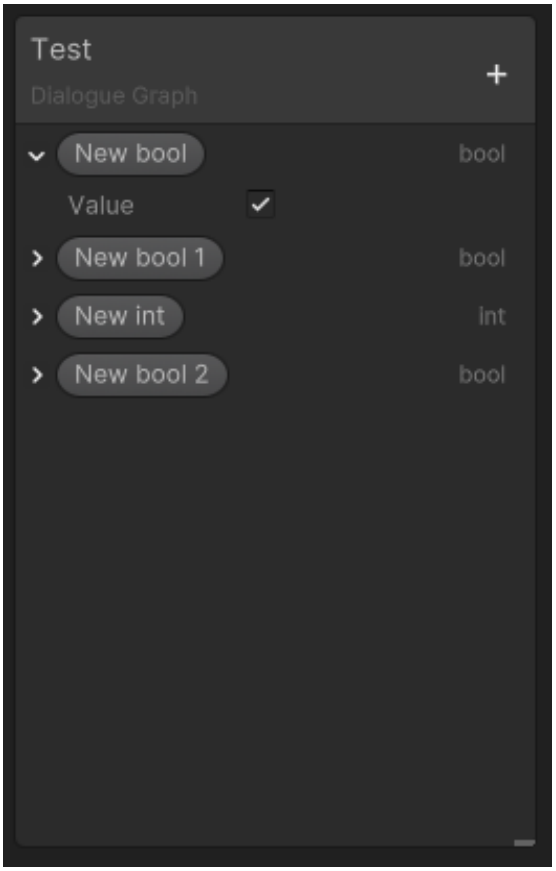

Fig. 5 Blackboard containing properties.

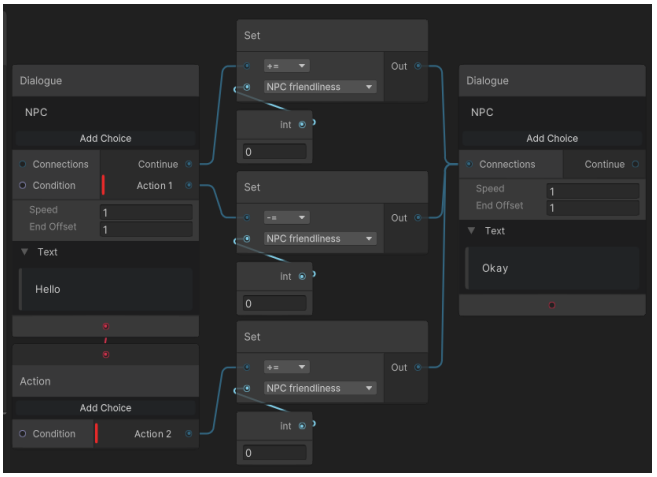

Fig. 6 Different ways *Set Nodes* can be connected.

The logic nodes are dependent on external properties, which are links from outside the graph that can change the NPC behaviour at runtime. These properties are constructed and arranged into a blackboard unique to that dialogue graph. There are different types of primitives that a property can inherit. Each property in the blackboard has a unique identifier accessible from code. The property also has a default value (e.g. Fig. 5) if the property is never assigned any value from the outside. The *Property Node* is connectable to the rest of the logic. The way the properties are modified during the conversation is through the *Set Node* (e.g. Fig. 6). The *Set Node* is used between the transition between two *Dialogue Nodes* or an *Action Node* and a *Dialogue node.*

To summarize the flow of the graph in Fig. 4, the player can take three different paths during the dialogue depending on the choices they make. If the player doesn't select an action at point 2 the path will be 1, 2, 11, 12, 13, where the player will have to select answer 1 at point 11. Selecting response 1(if it is available) in point 2 would result in the path: 1, 2, 4, 5, 6 and response 2 would lead to the path: 1, 2, 7, 8, 9, and 10.

1) Custom: The graph is expandable from custom scripts. Nodes and property types are addable and modular using the exact same system that all existing implementations use. If a logic node is missing, it is possible to create a new type of logic node and connect it to behaviour at runtime. It is technically feasible to repurpose the dialogue graph to a different type of graph.

The attributes connected to both objects (see Fig. 7, 8) register the new content to the graph. *CreateNode,* seen in Fig. 7, has two optional parameters, the first being the entry path and the second being the title name displayed on the node. *BlackboardProperty* seen in Fig. 8, has one optional parameter for a custom type name if the connected type's name is undesired. Both are registered using reflection.

As mentioned previously — the dialogue graph does not contain any runtime behaviour — rather, it defines the values and connections interpreted at runtime. The data is stored in a custom file extension called ".dialoguegraph". ".dialoguegraph" is the same as ".json". Because the actual data inside of the file is compatible with JSON, the data is human readable and even modifiable from the text file when debugging.

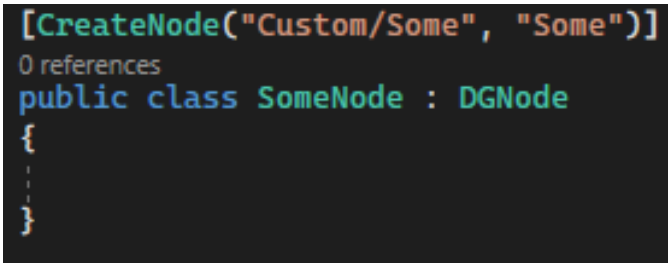

Fig. 7 New node type.

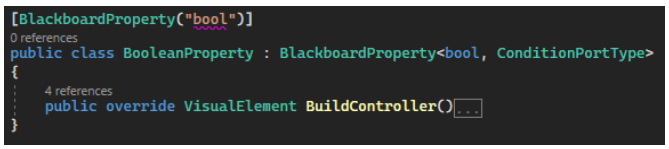

Fig. 8 Property of type boolean.

2) Reason: The reason for a graph editor is that it allows the user to create intricate connections between nodes that would not be feasible for a human to understand purely in text form. It would be possible to edit a JSON manually but would require manually dealing with identifiers for dependencies between dialogue snippets when the graph editor only requires the user to drag edges between ports from nodes. The graph editor compresses the different parts into compact nodes that contain all the information needed, while pure JSON would be more spread out into multiple rows for the same information. The graph editor has all the default values already set when creating a node and does not need to deal with missing data if forgotten to add all data for a node. The final reason is readability. JSON does not support multiline strings, which would result in longer and confusing rows of text. The deeper the dialogue goes, the more curvy brackets are used, which would lead to less and less information at hand. Dialogue with JSON is only reasonable when creating simple and linear dialogue without going too deep or having too long text.

D. Runtime

The runtime represents the dialogue system used in a game. The dialogue system and its parts consist of modular pieces, some of which function independently from the dialogue system itself. These modules are *Graph*, *Timeline*, *Interpreter*, and *Processor*. The dialogue system is the hub that links the data, such as the exposed properties created from blackboard properties from the dialogue graph editor. To deal with scene references — an external script can insert components such as the buttons used to express actions for the dialogue system to use when generating buttons to press. The dialogue system inside Unity is what is called a *Scriptable Object,* a persistent scriptable asset that lives inside of a game itself instead of being connected to a scene. The reason is that the dialogue system should persist seamlessly through different scenes at runtime. A dialogue could continue through multiple scenes without any intermission which means the current playing dialogue should not get affected. Unity Timeline (more in section 6.D.2) requires a director *Component* to play the timeline. *Components* are behaviour connected to a *Game Object* that lives in the scene. For this reason, the dialogue system (*Scriptable Object*) instantiates a *Game Object* with the necessary components every time the current scene has changed. The instancing is done at playtime and does not need any pre-instanced objects in any scenes other than the required UI elements to display the dialogue.

1) Graph: Creating dialogue requires sufficient data to express the flow in which nodes defined by one or multiple behaviours are capable of being interpreted. Having the necessary information expressible in such a dialogue is only one part of the equation. The dialogue editor previously mentioned generates such data readable from a text file. These text files do not possess any logic capable of expressing a functional dialogue but instead building blocks used to reconstruct a runtime graph that only in itself is for cohesion purposes. This runtime graph is deserialised and reconstructed from the text file. The graph contains a specialised dictionary containing all Input Nodes from the editor. It also contains a dictionary with all backboard properties that are linked by events to corresponding Value Processor (see section 6.D.3) nodes in the graph.

A runtime graph contains nodes. Nodes consist of ports, and ports consist of edges connecting different ports resulting in nodes connected to other nodes. Edges are containers containing references to one input and one output port.

Because nodes can have multiple ports, the index needs to be in the correct order, similar to how they connect in the dialogue graph editor. Vertical ports have a negative index, meaning ports for a node are stored in a dictionary where the index is the key. The order is crucial because different output and input can possess various characteristics. In Fig. 4, there are different types of nodes containing ports. For example, at number 3, the *Condition Node* has two inputs and one output. The inputs in this context are numerical ports, meaning it will take the numeric value returned from a child node. The output is the resulting value processed and returned as a boolean. Port *A* and *B* have different purposes in the equation and need the correct order at runtime. The "Greater Or Equal" compares the *A* and *B* values with each other. In this case, it checks if *A* is greater or equal to *B*. The previously mentioned "Greater Or Equal" field is a local value connected to the node. The node in the editor is purely a representation of what information the runtime counterpart entails. The runtime node nor the editor node executes any behaviour; instead stores information created from a text file that contains the linked binder type to generate the behaviour when needed.

When playing a dialogue, a reference to at least one node is required. In this case, the start node is the *Input Node* seen in Fig. 4, node 1. All runtime nodes possess a binder type (behaviour type) and values linked to that node. On dialogue generation, the binder type instantiates and assigns the values from the node. There are two different types of binders: *Interpreter* and *Processor*. The *Interpreter* contains the behaviour to create corresponding logic. It can instance the binder from connected nodes, which in turn instantiates its binder from connected nodes. *Processors* store and evaluate content to connected clips in a timeline (see section 6.D.2).

2) Timeline: Timeline is linear data where the index represents a time in space. This data is called a track. It uses Unity Timeline, which allows the user to create custom tracks. Tracks can coexist parallel to each other and share the same time in space seen in Fig. 9. Tracks have a start and endpoint. A timeline can have sub timelines inside of itself. A timeline is playable, which changes the current time in the timeline. In Fig. 9, the white vertical line shows the selected time. It can be manually moved or played from the top-left buttons. Tracks contain clips and markers that occupy space on a track. The clip is a slice of the track with a start and end. It activates when the current time is inside the clip range. In Fig. 9, the selected time is inside two clips on two different tracks. Similar to the clip, the marker activates when being passed. The difference between a clip and a marker is that markers represent one point in space. Multiple markers can coexist at the same point.

In Fig. 9, there are three different types of tracks: *Dialogue Track, Action Track,* and *Signal Track.* The *Signal Track* is a pre-existing track type purely for markers of all kinds. It is possible to place markers on other tracks, but for readability purposes and not being related to any of the other tracks — the use of a separate track is the preferred solution. The dialogue can invoke events during the play that translates to the *Signal Emitter* markers seen in Fig. 9 as blocky white arrows pointing downwards. An asset existing on a timeline does not in itself exist inside the scene. Because of this, the marker is referenced to an event receiver component connected to a *Game Object.* This receiver is the one which holds all the events and can access objects and components in the scene.

Dialogue Track has dialogue clips which contain the paragraph text, actor and the deliverance settings for that paragraph. The clip length determines the duration the dialogue should be displayed; This, in turn, is read through events that the user can subscribe to. The dialogue events are called from an intermediate component that binds the timeline clip with scripts in the game. This solution adds complexity and could possibly be moved to the dialogue system for a centralised access area.

The *Action Track* deals with all interactive elements in the dialogue. One supported feature is linking a clip to a UI button. The clip length corresponds to the duration of how long the button should be available to be pressed. It contains a button reference, condition (see section 6.D.4) and action settings. Action settings are a *Scriptable Object* which currently allows the user to change how the button should fade, meaning the visibility. In Fig. 10, the setting has sub-fields consisting

of modifiable curves. The x-axis is the duration, while the y-axis is the visibility. Visibility ranges from 0 to 1. Because the action settings are a *Scriptable Object* — which is referenced in the clip — can be shared and changed from one centralised source. Another possible feature would be to support individual references for every field in the settings. Override, similar to how *Prefabs* in Unity operates, could possibly make the user be able to fine-tune anything they desire to change just for that clip in the timeline. The condition seen in Fig. 10 determines if the button is active or not. This mechanic allows for options that are only available based on external factors, such as you have reached a certain friendliness value and can access more dialogues as a consequence.

Another action clip is the *Action Event* which works similar to the *Signal Emitter* whose event is stored on a receiver component in the scene. As seen in Fig. 11, the inspector shows the event connected to the asset, but the event itself is stored in the component and not in the asset itself. What differentiates it from the *Signal Emitter* is the range. Instead of being a point in time, the *Action Event* exists between a start and end time. This is useful when dealing with an enter and exit situation.

The reasoning as to why the timeline is used to express the flow of the dialogue is that it already has support for multiple clips simultaneously playing. It can activate and play different clips asynchronous to one another which previously explained is needed to express interruption. Though the current generation of the dialogue does not give the option for intermediate choices, it would be feasible to implement it. The current generation maps the choice to the length of connected dialogues, meaning the dialogue length determines the duration of that choice. Another reason is that it already has an interactive window that allows manual debugging of the different clips without the need of being generated. It can also be viewed and modified at runtime. It is possible to see the current runnable timeline connected to the dialogue system. If something did not execute as expected, it would only be needed to inspect the timeline for any anomalies. The last reason is that it allows for modularity. The dialogue system is not required to use the available dialogue behaviours and can be used independently or with another dialogue system implementation. The one provided is not a requirement for the dialogue to work. The timeline is also the way to create linear cutscenes in Unity. Having equal implementation for both conversational and cinematic situations allows the designers to integrate the same tools and use a standardised workflow. Any changes made to one of them will affect the other. Not all conversations will have multiple choices, and the designer should use the tool that does the work fastest. Creating the dialogue directly inside the timeline might be the quickest solution in one workflow.

A downside to using a timeline is how multiple paths should be handled. In the current implementation — when generating a timeline — each time the player chooses another path (for example, by pressing a button), the timeline regenerates with the next node being the start position. The timeline generates all possible choices and dialogues for that path, meaning if it is possible to not change its course through the dialogue, the whole dialogue from the beginning to the end is created.

Another solution would be to generate clips and markers when needed in front of the current time. When one clip is soon due, another clip is created in front of the current one. Unity Timeline — the timeline implementation used — does not support changes to a timeline without regenerating it. Because the timeline is immutable, generating everything at once and when taking another path was the more suitable solution. Generating each time a new path is chosen results in reallocation, but only when changing course, which results in infrequent changes.

The current weakness is the intermediate blend between the previous timeline with the new one. It does not support the possibility to blend any clips with each other nor continue an already existing clip that extends past the old timeline's current time when changing the path.

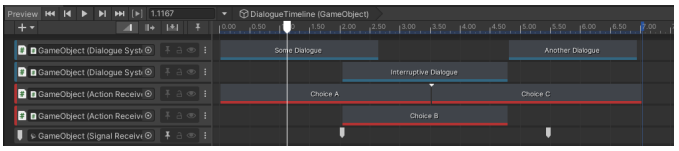

Fig. 9 Timeline containing different types of tracks. An example of interruption, button choices and markers.

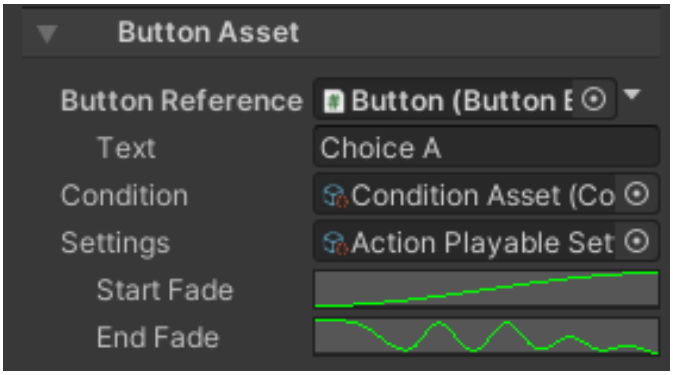

Fig. 10 Button Asset. Contains information about the button clip.

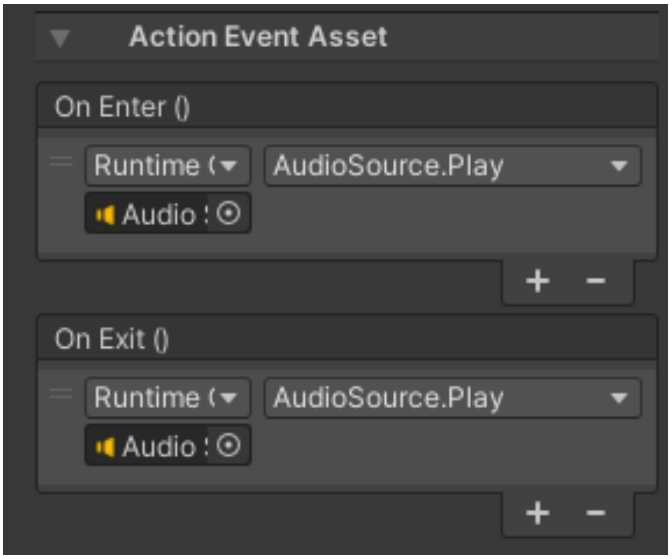

Fig. 11 Action Event Asset. Binds to an action receiver which stores all events.

3) Interpreter: Interpreter is an object linked to a node. It interprets the connections and values from the said node to create the intended behaviour of the editor node counterpart. The interpreter is the behaviour that generates the dialogue and other necessary parts. What it is capable of doing is up to the implementation.

The currently implemented interpreters are:

- Action Interpreter
- Blend Interpreter
- Dialogue Interpreter
- Input Interpreter
- Marker Interpreter
- Set Interpreter

The interpreter for the *Dialogue Node, for* example, creates a dialogue clip and button clips when any is available. If no track of that type exists, it will create a new track or if the existing ones do not have the space desired at that time position. It will also interpret any *Action Interpreters* connected to it from its vertical output port (see Fig. 4, number 2 where the vertical output port is connected to number 7's vertical input port).

Interpreters are modular and self-sufficient. It will check if the condition for this interpreter is valid or else do nothing. An interpreter will only be instanced when needed, meaning it could be when choosing a path on which it locates.

Another interpreter is the *Blend Interpreter*. The *Blend Interpreter* does not create new clips but rather by modifying the existing ones. An interpreter has two methods: OnInterpret and OnNext. OnInterpret does the behaviour intended for the interpreter while OnNext is when it is done and moves to the next node. The *Blend Interpreter* utilises both methods. It searches for any *Dialogue Interpreter* in front of itself. If found, it will instance the interpreter and run its OnInterpret which creates the clips. After being done, it modifies the start and duration of the clips. Because blending means moving a clip/clips closer in time, sometimes there will be overlap. If overlap occurs, it tries to find another space at the same time position, but instead on another track. If no track was found, it will create a new one and place itself in the new time position. The blending is how interruption occurs. After completion, it calls the OnNext on the *Dialogue Interpreter*.

Interpreters are vague in their capabilities and could technically be used to generate something other than a timeline, such as a traditional state machine. They are intended to modify other interpreters who are unaware of its existence. The processors are also created by interpreters who have a connection to them. This will call on all linked processors in the tree. If a processor already exists, for example, when connected to two different locations, it will reuse the already created one and continue down the tree (see section 6.D.4).

4) Processor: Processor is a Scriptable Object that handles input port values which become processed and

returned from the specified output port. They can be used for conditions when determining if a button should be displayed or not. Similarly to Timeline, it can work independently and be used for things other than dialogue. When initialised, it will initialise all its children recursively.

The currently implemented processors are:

- Value Processor
- Condition Processor

A processor has only reference to its children and does not know about its parents or if it has any. When evaluating, it will recursively go through all its children before processing their values that later becomes processed by its parents if it has any and repeats.

Value Processor contains one value of the preferred type. When accessed, it returns the value. *The Condition Processor* takes two values — A and B — and compares them to one another based on the compare enum. It supports all C# compare types such as equal, not equal, greater or equal etc. It returns the result as a boolean upstream. *The Condition Processor* can be used to check when a friendliness value has reached a threshold.

The flow of data is always moved upstream, but this poses a challenge when a value down the stream changes. A value from a *Value Processor* only contains a value and returns it when changed. It will call itself dirty on change which activates an event to its parents. When processors are initialised, the parent subscribes to the child's event, which invokes when the child processor is dirty. When the child is dirty, the parent also becomes dirty. When a processor is dirty, it will continue up the stream until reaching the owner who uses the value. This implementation means it does not need to check each frame if any change has been made and only recursively re-evaluate when required.

5) Emotion: The part of the system that manages emotion is inspired by previous research [1, 2, 3] where they made the emotions of the NPC dependent on the player's actions. The most common way is to identify the actions as positive or negative before the NPC registers and updates their current emotional values according to the input. The system tracks the emotional values of the NPCs which their emotional state depends on, similar to a dialogue system from another research called EDTree [3], which stored emotional values and used them to make a player action able to produce multiple different NPC responses depending on the values. Each NPC holds emotional values (friendliness, upset and others that the designer might set) that get affected depending on the interaction with the player. The NPC stores its own emotional values individually, meaning that only the NPC interacting with the player will be affected by the chosen actions. The reply alternatives that the player picks during dialogue with an NPC will influence one or multiple emotional values of the given NPC by increasing or decreasing its value. The magnitude and the type of each modification are based on the player's action (each modification that an action has, is set by the dialogue designer). By implementing emotional values, it is possible to give the illusion of emotion and relationship by locking and unlocking player activities with a NPC depending on the emotional values of the corresponding NPC.

In our implementation, we have defined 2 emotional values, which are friendliness and upset. Instead of having one player action having the possibility to produce multiple NPC responses depending on the emotional values we use a different way to produce emotional NPC responses. To give the illusion of emotion, the emotional values were used to unlock or lock specific player actions that had emotional outcomes. By only unlocking an action that leads to a path where the NPC involved responds emotionally after the player has modified an emotional value enough through their previous actions, it is possible to stimulate emotion. For example, being rude to the NPC will unlock the actions which results in the NPC responding with an angry dialogue while locking the actions that have a friendly response.

VII. RESULTS FROM USER STUDY

Our main hypothesis is that the implementation of interruption in conversations within games will increase user engagement with the NPC and with the story compared to the traditional turn-based dialogue system.

When the participants were going to choose their favourite version of the game, 7 out of 8 picked INTERRUPTION. A trending answer the participants gave when asked why they picked INTERRUPTION as their favourite was that they felt it was more interactive and immersive since they had to pay attention to the present state of the conversation and make decisions in the heat of the moment similar to how it is in real-world conversation.

Three-quarters of the participants also thought that the INTERRUPTION version of the game offered the player more choices.

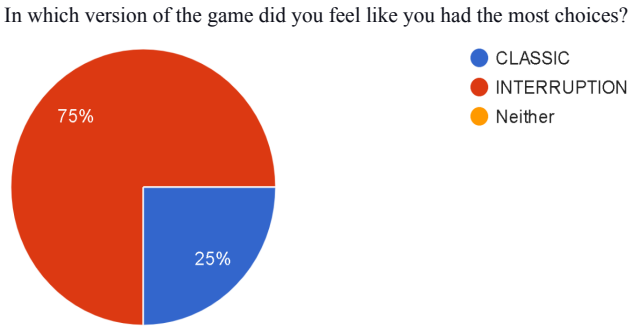

Fig. 12 Result from participants.

All of the participants except one, thought that the CLASSIC version was easier to use. When asked about the reason, they either answered because they were familiar with that kind of dialogue system in games or that it gave them more time to think about their actions and their possible outcomes compared to the INTERRUPTION version. Three of the participants thought that the speed of the dialogue was a bit too fast and that they barely could keep up. Only one participant thought that INTERRUPTION was easier to use and answered "*Interruption, because it engages you more and makes you think, and triggers your brain to take fast action.* " when asked why the participant thought the INTERRUPTION version was easier to use.

The results from the questions comparing the two different game versions with each other through questions with a 5 scale answers option can be summarized in these tables:

In table 1, the participants either picked that they were more engaged in the INTERRUPTION compared to the CLASSIC version or that they were as engaged in both versions. When asked about the reason why they picked as they did the answer was that it's because you are under time pressure to make decisions compared to the CLASSIC where the NPCs would wait for your answer, which forced them to pay more attention to the conversation. One of the main reasons why the participants found INTERRUPTION more engaging was also because they thought it felt more like a real-world conversation.

In table 2, the results from the participants were a bit mixed, split between those who thought that they could affect the story more during the CLASSIC version because they had more time to think through the decisions and those who thought they had more control over the story in the INTERRUPT version since they believed that both the actions and the timing of the interruptions impacted the story. But the table shows that more participants felt they could influence the story more in the CLASSIC version.

When asked what strategies the players used when picking the different options and actions most of the participants answered that they picked differently based on the version of the game. In the CLASSIC version, the majority of the participants said they selected the action that they thought would have the most interesting outcome.

TABLE I
HOW ENGAGED WERE YOU IN THE DIFFERENT VERSIONS OF THE GAME?

	1	2	3	4	5
INTERRUPTION	0	0	1	1	6
CLASSIC	0	3	2	2	1

TABLE II
HOW MUCH INFLUENCE DID YOU FEEL YOU HAD OVER THE STORY IN THE DIFFERENT VERSIONS?

	1	2	3	4	5
INTERRUPTION	0	0	3	3	2
CLASSIC	0	0	3	1	4

But during the INTERRUPTION version, the answers varied with small similarities.

Below are some examples of what the participants answered:

"*During interruption, since there was no way to pause and give myself time to think it made me more prone to choose options based on intuition rather than through strategizing, which led to making decisions more in line with as if I were there myself.*
During classic, since I was in complete control over when to proceed I had more time to think and plan of course, so I chose options that sounded more interesting.".

"In the classic, this happened naturally at my own pace. Whereas, in the Interrupted version, I had to act fast and thus think fast, so my decisions were directly corresponding to the engaging environment of the dialogs."

"*First gameplay was pretty straight forward, being on the snakes side. Second gameplay was more about how i can mess with the buttler and the snake.*".

During the verbal interview when participants were asked if they could feel any lasting consequences of the interactions between the player and NPCs during INTERRUPTION, half of the participants mentioned that they felt like their actions had lasting consequences.

"*I felt like it affected the story in the long run because I made Madam very angry at the end and she kicked me out"*

"Yes it did! I think that because of the expectation of waiting longer for more options increasing the risk of negative consequences I was more likely to take a polite option sooner as to not upset the lovely snake. It at least felt like the options I ended up choosing changed the course of the conversation due to the different responses I was given, especially when compared to the options I chose the second playthrough."

"Yes, especially on my 2nd gameplay when i tried to mess with both of them the dialog changed completely compared to my first gameplay."

"I kind of feel, that, it went on for much longer. In addition it had a triggering loop effect (kind of ripple effect), where both NPCs were suddenly involved in the conversation. It also produces a more lasting effect, because you remember this afterwards as a player. I like the Interrupted version, because of the positive engagement."

While the other half didn't notice any difference in consequences in the different versions.

"*It felt the same as the Classic mode when you compare the consequences. Although, it felt more interactive with the interruption mode, the feeling of the consequences did feel more immersive than the classical."*

"*Interruption mostly felt like a more interactive version of classical.*"

VIII. DISCUSSION

In this section, the data collected from the study will be discussed with regard to the aforementioned research questions. The limitations and weaknesses of the study will also be addressed during the discussion.

The data collected shows how interruption during the dialogue has an effect on user engagement in the used game scenario. The participants rated the INTERRUPT version of the game as more engaging with one of the reasons being it puts pressure on the player to act and react in the heat of the moment, similar to real-world conversations. When asked which version the player felt had the most choices, the majority also said INTERRUPTION, which is interesting since the different game versions had the exact same player options and dialogue routes; this supports the notion that implementing the ability to interrupt in video-game dialogues may cause dialogue feel more engaging for the player and gives them the illusion of having more choices in some scenarios.

One conclusion from the results is that the ability to invoke emotions gives the users the feeling of lasting

consequences from the actions rather than the different dialogue paths that they could explore from their choices. Almost all answers that believed that the decisions made during the INTERRUPTION version had lasting consequences contained parts related to the NPC's emotions. The participants stated that they tried to *mess* with them or not tried to make them *upset*, which indicates that they believed that they could invoke emotions from the NPC when there were none in the game scenarios. It is plausible that interruption enhances the illusion of emotion.

The results suggest that there is potential in the new proposed approach, which combines the ability to interrupt other actors mid-sentence. According to the study, the participants thought that they had more influence over the story during the CLASSIC version due to having more time to think through their action options and the possible outcomes before deciding what to do. But the participants preferred the INTERRUPT version of the game, despite the fact that they felt like the CLASSIC version gave them more control over the story and was easier to operate.

The study had a number of limitations and weaknesses which could have possibly changed the final outcome. The biggest weakness was the limited number of participants during the tests consisting of only eight people. Having such a small group makes it difficult to draw accurate general assumptions with the possibility that they are all exceptions that deviate from the norm. Another limitation of the study is that the player could not change the speed of the revealing of the dialogue text, which made some players barely able to keep up with the pace of the text. The results of how much influence the players felt they had during the different versions might have been affected due to some of them not being able to read the dialogue and keep track of the actions appearing, therefore making them feel like they had less control over the story. A weakness of our study is that the game did not implement the emotional values even though it was possible with our artefact due to time constraints. There were endings with dialogue simulating emotions, but the paths to reach the endings did not take advantage of the emotional values making this conducted research focus more on interruptions rather than interruption and emotion.

Because the participants' first experience is with one of the variations, the participants would experience the game as a whole for the first time. The first time playing resulted in playing more safe and not experimenting with the choices, which could affect the final result. Because of already playing it once, the second time would be with existing experience and trying out other options knowing what consequences their previous playthrough resulted in. Another thing that needs to be taken into consideration is that with the INTERRUPTION version, the game is more complex, and playing from INTERRUPTION to CLASSIC could be seen as a downgrade by some, which could affect the overall enjoyment of that play section. It may be a better experience not knowing what is missing rather than knowing; further research is needed to validate this.

The testers never saw a full image of the consequences and what would lead to what during the test; this was to keep the illusion of the game having more depth than it had and to make the testers think and anticipate the consequences of each action similar to how it is in real-life conversations. It could be an intriguing topic to study in more depth how the illusion of choice affects the enjoyment of the experience. Another

The study established that our novel approach to handling dialogue could potentially improve the conversational experience for the player due to the positive results regarding enjoyment, engagement and the perceived amount of player choice.

Having asynchronous dialogues in games has further potential — giving an illusion of more player choices and the possibility of new mechanics in the game. An asynchronous dialogue system makes it possible for player options to appear at any time as well as making the same player option have different consequences depending on the timing the player chooses the option; this may encourage the player to be more attentive to the current state of dialogue and make dialogues feel more part of the gameplay instead of purely be a mechanism to deliver information to the player when needed. Having additional features beyond pressing continue and, on occasion, different choices could require the player to engage with the dialogue in another way. Additionally, it also supports two different

dialogues to run parallel to each other. Having the possibility for two separate dialogues to run simultaneously may make the dialogue feel more dynamic for the player and enhance the gameplay experience and immersiveness. These new features that our asynchronous dialogue system has implemented can possibly contribute to a more engaging and immersive conversational experience compared to the standard dialogue system.

The current implementation of the graph editor currently only supports two participants speaking simultaneously. The solution was to create a node (*Blend Node*) connectable between two *Dialogue Nodes*. It allowed the developer to set a blend duration resulting in the two dialogues playing simultaneously during that time window. To deal with more than two participants, a new type of node (or something else), which does not need to replace the *Blend Node,* to solve multiple participants speaking. The rest of the system does support more than two participants at once, and how that should be expressed as nodes is the only thing uncertain. One solution is to split the different dialogues by participants into different paths. Instead of being paths affected by player action, it executes parallel with other branches. The challenge which results from branching the dialogues is the time aspect. The current *Dialogue Node* does not in itself give any feedback to the developer on how long the duration is. One possible change would be to have the width of the *Dialogue Node* indicate the length of the actual clip. *Dialogue Nodes* would be internal timelines inside of the graph editor. The execution time would be readable even with multiple branches playing simultaneously. Other changes would need to be made to accommodate this change. For example, the *Dialogue Nodes* should be snappable to each other and better, not require a connection between each other by edges to ports. An interface more similar to how the timeline is handled in figure 9 but as a node/group placable in the graph that can have a connection to other nodes — combines the ease of use of a timeline with the non-linearity of a graph. Multiple timeline nodes/groups should be playable concurrently allowing the developer to change based on outside factors what should be played.

The appreciation of the previously explained extension compared to a traditional dialogue graph is that creating asynchronous dialogue is more complex and harder to read. There are different ways to deal with it, but because it revolves around additional features, it in itself will always be more complex. The current use of *Blend Node* is manageable because it works similar to other nodes and can be opted out of if a non-interruptive conversation is desired. It constraints the complexity to itself which would be ideal for any feature to not affect other parts of the graph. It is another node to deal with, which results in more nodes in the graph. The new feature mentioned with the internal timeline would be an additional optional feature for more intricate dialogues. That asynchronous dialogue is more complex than traditional does not mean it is a downside. The wish to create more complex scenarios results in needing more complex information. The editor is unable to know what the developer wants if not expressed explicitly.

IX. CONCLUSION

This study confirmed that introducing interruption in dialogue might enhance the player experience by making it more engaging and enjoyable. As a consequence, some became more stressed and less confident in what to choose; this, by some, is positive because they are put in a more pressured situation resulting in the experience feeling more immersive. Conclusions that the illusion of players being able to evoke NPC emotions makes them feel like their actions and decisions during interactions with NPCs have long-lasting consequences can also be drawn based on the results.

Because of the scope, further research on interruption is required, both the sample size and also how interruption in voiced conversation affects the experience for the player. A longer game that takes advantage of both interruption and emotional values could supposedly study how those correlate and what effect they have on the experience.

For the future of this dialogue system, its development will be continued on and may, in the foreseeable future, become a proper package that is downloadable. The intention is that the dialogue system

will be used in future projects. Any decisions tailored specifically for this project may be revised.

REFERENCES

[1] J. Fraser, I. Papaioannou and O. Lemon, "Spoken conversational AI in video games – Emotional dialogue management increases user engagement", *Proceedings of the 18th International Conference on Intelligent Virtual Agents*, IVA, 2018.

[2] M. Metha, K. Mishra and A. Corradini, "Dynamic relationship management for personality rich character presentations in interactive games", *Proceedings of the 5th Conference on Speech Technology and Human-Computer Dialogue*, SpeD, 2009.

[3] J. Collins, W. Hisrt W., W. Tang, C. Luu C, P. Smith, A. Watson and R. Sahandi, "EDTree: Emotional dialogue trees for game based training", *Lecture Notes in Computer Science(including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016.

[4] D. Rhee, I. Won, H. Song, H. Park, J. Chang, K. Park and Y. Cho, "A 3D-dialogue system between game characters to improve reality in MMORPG", *Lecture Notes in Computer Science*, vol. 3332, 2004.

[5] J. Ryan, M. Mateas and N. Wardrip-Fruin, "A Lightweight Videogame Dialogue Manager", *Proceedings of 1st International Joint Conference of DiGRA and FDG*, 2016.

[6] B. Grosz and C. Sidner, "Attention, Intention, and the Structure of Discourse", *Computational Linguistics*, vol. 12, 1986.

[7] S. Domsch, "Dialogue in Video Games", *Dialogue across Media*, edited by Mildorf, Jarmila, Thomas, Bronwen, 251–70, Amsterdam, Philadelphia, PA: John Benjamins Publishing Company, 2017.

[8] J. Brusk, "Game Dialogue Management", 2006.

[9] H. Sacks, E. Schegloff and G. Jefferson, "A Simple Systematic for the Organisation of Turn-Taking in Conversation", *Language*, vol. 50, Is. 4, 1974.

[10] Peter French and John Local, "Prosodic Features and the Management of Interruptions 1", *Intonation in Discourse*, 1986.

[11] Li-chiung Yang, "Interruptions and Intonation", *International Conference on Spoken Language Processing*, ICSLP, Proceedings, 1996.

[12] J. Bates, "The role of emotion in believable agents", *Communications of the ACM*, 37(7): 122125, July, 1994.

[13] K. Peffers, T. Tuunanen, M. A. Rothenberger, S. Chatterjee, "A Design Science Research Methodology for Information Systems Research", *Journal of Management Information Systems*, 24:3, 45-77, 2007.

[14] S. Sali, N. Wardrip-Gruin, S. Dow, M. Mateas S. Kurniawan, A. A. Reed and R. Liu, "Playing with Words: From intuition to evaluation of game dialogue interfaces", *FDG, Proceedings of the 5th International Conference on the Foundations of Digital Games*, 2010.

DIALOGUE SCENARIO

Madam: An interest in convincing the player to support her faction at all costs.
Charles the Bug Butler: A loyal servant of Lady Madam.
Player: Of great importance. The reason is unknown.

- Greetingssshh, my honourable guessht.
- Charleshh pleashse ehscort the guessht to the garden,
- we have important matterss to disshcussshh.

**walks to the garden **

- Honourable guest, please have a seat while I
- prepare some refreshments for you and Lady Madam.
- Glorious weather today, isn't it, Lady Madam.
- Oh yes... we haven't had this sunny in a long time.
- …

*-> Main

Action 1.1 - NO I'M GOOD

- I think I'm good.

Action 2.1 -> Action 1.1

- Sshhh no-no, pleasse make yourshelf at home.
- Charleshh! Don't sscare away our honourable guessht.
- I need to apologisshe for hiss mannerssh.
- I don't know what ish up in hiss buggy head.
- Charlessh! Apologisshe for your ill mannershs!
- I- I am terribly sorry. My greatest apologies.
- Ho- Honourable guest,
- is there anything I can possibly do for you?
- Ju- Just ask, and I will grant it to you.
- I might look like a bug,
- but that doesn't mean I am just bugging around.
- …
- Look, Charlessh, you are ssharing the guesst.
- I knew it!

Action 1.1.1 - *DE-ESCALATE*

- I- I didn't mean that way.
- Oh — my apologiess — in that casse,
- let uss continue our conversshation.

Takes a seat

- I hope you like my modesst mansshion.
- It is high up and dessolated from everything elshe.
- That iss how I like it.

Action 1.1.2.1 - YES

- I would lik-
- What you need iss shome brewed tea. Right Charlessh?

- Yes, Lady Madam.

Action 1.1.2.2 - NO

- No, it is fine.
- Good, then let us continue the conversation.

Action 1.1 -> Action 1.1.2.2

- Charles, make yourself useful and bring the tea.
- Yes, Lady Madam.

1.1.2.1 -> Action 1.1.2.2

- I promise that you will love it.

Action 1.2 - *Sit*

*-> Main

Action 2.1 - TRY LEAVE

- Well, look at that. It seems like I have something else to attend to.

Action 2.2 - I'S PRETTY

- Do you really think so? I am truly flattered.
- Charles is a masterful gardener.
- I guess it is in his roots tthssss tthssss.
- Yo- you don't need to flatter me Lady Madam.
- Your present is honour enough already Lady Madam.
- Oh, Charlesshs.
- Honourable guest, Lady Madam is well known for her excellent tea making.
- Oh, Charlesshs.

Action 3.1.2 -> Action 2.1

- Speaking of which, I almost forgot the tea.

Action 2.2.x -> Action 2.1

- Excuse me for a moment.

Charles returns with a plate with a teapot on top

Action 2.2.1 EHEMM

Action 2.2.2 - *LOOK AWAY*

Action 2.2.3 - *COUGH*

- Oh thsss thsss.
- My Apologuessh.
- Were where we.

Action 3.1 - THE BACKDOOR

- Th- the backdoor? Charlesshs?
- CHARLESSSHSS!
- You didn't forget to clossse the door again, didn't you?
- La- Lady Madam, I- I…
- Charles! The last time you forgot,
- my rosseshs were eaten by those peshky rabbits crawling beyond the wall.
- My apologieshs honourable guesst. Charlessh will attend to it immediately.

Action.3.3.1 -> Action 3.1

- Isn't that right, Charlesshss?
- Yes, Lady Madam.
- I guesshs there is no other choishe than to sserve uss myself. ssshsss.

Lady Madam returns frustrated with a plate with a teapot on top

Action 3.1.1 - MY FAULT
- I should have closed it when I-
- NO NO don't blame it on yoursshelf.
- Charleshs will attend to it immediately.

Action 3.1.2 - I CLOSED
- You did? Well, if that is the casshe…
- Apologieshs for my tantrum.

Action 4.1 - NOT REALLY
- Not rea-

Action 4.2 - I-
- Well, I-

*-> Main

- What a pleasssant suprisse that you would accept my humble invitatissson…
- to my modesst ssummer manssion far away from everything.
- I hope you like my garden.

1.1.1-> Main
- I'm not ssure how you even came up here on your own,
- but regardlesssh.
- It might not be everyone'ssh cup of tea,
- but I persshsonally love nature and what it bringsshh.
- Shhpeaking of tea, CHARLESSHH —
- you are not sshlacking off, aren't you?

1.1.2.2 -> Main
- My apologies lady Madam — the tea is freshly brewed.

2.2-> Main

3.1 -> Main

pouring the teapot into two elegantly decorated teacups
- Anyway. I have a great fascsshination with you.
- I can sshee it in your eye.
- You dessire power,
- don't you?
- I knew you were shomeone worthy enough for my attentisson!
- I can already ssee the future ahead of uss.
- Don't you agree Charlesshs?
- Ye- Yes Lady Madam. I can see it.
- Am I talking too fasst?
- My apology. I haven't even assked if you want sshome sshsugar in your tea.
- Would you like a few sshsugar cubessh in your tea?
- No need to sshkimp.
- We got plenty.
- …
- Honourable gueshst?

- …

*-> Main 8

Action 5.1 - YES

- Extravagant tasste, just assh mysshelf.
- Charlesshh serve the guest!
- Yes, Lady Madam.
- And two more for me. thhs thhss.
- I am on a diet.
- They sshay everyone got their poisson, and mine iss the sweetnesshh of ssugar.
- Sshhss ssshhss ssshhss.

*-> Main 8

Action 5.2 - NO

- No sshsugar? Ssometimessh a bit of sweetnessshh is needed in life.
- Charleshs alwayss prefers hiss that way too. Issn't that so Charles?
- That is true, Lady Madam. I've always been sensitive to too much sugar.
- It might even help you sweeten up a bit, sshhss ssshhss sshss.

*-> Main 8

Action 6.1 - GREAT

- It tastes grea-

Action 6.2 - EHMM

- It ehmm-

Action 6.3 - *GROWLING*

- I- I think I nee-

- How doess the tea tasshte?
- Deliciouss right?
- It's black tea, and I ussually add shsugar and mint into mine.
- Tea hass cultural importanshe to uss in this land.
- But it hassh been shhallenged with the recent introdusstion of…
- that nassty black water called coffee.
- I asspire to pressherve our opulent mannerss and distanshe from…
- what deviatess the tasste budshs of the masseshs.

Action 7.1 – INVITE CHARLES (charles friendliness +2, madam upset +1)

- Charles, why don’t you join us at the table?
- …
- S- Surely you jest, I need to follow proper etiquette…
- if I want to continue to serve as Lady Madam’s Butler.
- I would never dare to tarnish her good name.
- I hope you understand the reason why I have to decline your generous offer.

blend madam and charles

- Shtop bothering our guessht Charleshh

*-> Main 8

Action 7.1.1 - JOKE

- Hhehhe it- it was only a jest…
- Thhhsh thhshhs.
- Oh, honourable guesst, I didn't know you were sussh a joker.
- …

*-> Main 8

Main 8

- Sho ash you know the Royal sshelection where…
- we decssde the monarch of the kingdom ish shoon due.
- And I'm looking for sssupport.
- Meaning you.
- Tell me, do you like BUGSSH?
- Yes or no question.
- …
- It issh pretty rude to ignore sssomeone…
- when they are ssspeaking to you.
- Well, I myself love bugs.
- They have sshuch a refined and exquisshite tasshte.

*-> YES 8.1.1

Action 8.1 – …YES (Charles friendliness +1)

- Yes, I think they a-
- Well, I enjoy them assh well!
- They have sshuch a refined and exquisshite tasshte.
- That isn't what i mea–
- *-> YES 8.1.1
- We are pretty ssimilar don't you think?
- …. +

*-> Main 9

Action 8.2 – …NO (Charles friendliness -1, madam friendliness +1, upset -1)

- Not really.
- …
- How bold of you to sssstate sssuch a thing in front of Charlesssh.
- Courageousssh even!

*-> Main 9

Action 8.2.1 SORRY

- Charles, I didn't mea-

*-> Main 9

Main 9

- Thiss issh why I want you on my sside when I become the monarch.
- I wass already fashcinated by you before but...
- now I'm intrigued by your perssonality.
- Ssupport me during the Royal sseleshtion and I…

- will create the shountry that you desssire.

Action 9.1 – REJECT (madam friendliness -2, upset +2 due to interruption)

- I'm sorry but I'm not really interested.
- Yo- You don't care?
- How can you not be? Do you not care what…
- happenssh to anyone or anything?
- Then hear me out before you make your decisshion,
- These matterssh are of great importance!

*-> main 10.0

Action 9.1.1 - NO I DONT

*-> Mad rejection end

Action 9.1.2 - I DO

- I do bu-

Action 9.1.3 – KIND REJECT

- I'll have to respectfully decline your offer…
- I do wish you the best of luck.

*-> neutral rejection end

Action 9.1.4 – TELL MORE

- Sure, I will listen to what you...

*-> Main 10.0

No action 9.2 (madam friendliness + 1 since no interruption)

- Our nation might be great but I ssstill believe…
- that there issh much to improve.
- Therefore I want to reform the kingdom.
- By doing ssmall thingssh,
- I believe we can achieve greatnessh!
- Sso what do you think about the proposshal?
- …
- …

waits for player answer **interruption possible**

Action 9.2.1 – *HUMBLY DENY*

- I'm happy that you think so highly of me…
- but I'd rather not get involved.

No action 9.2 -> Action 9.2.1

- Could you consshider it?
- …
- …
- …

*-> neutral rejection end

Action 9.2.1.1 – NO
*-> Mad rejection end

Action 9.2.1.2 – YES
*-> Neutral rejection end

Action 9.2.2 – *UNSURE*
- Could I have some time to think about it?
- As long ash you consshider it I'm happy.
- This has rather been a pleassant day hassn't it Charlessh?
- It certainly has Lady Madam.
- Sorry to intrude but Lady Madam's…
- next appointment is soon due.
- OH MY! (**Blend this part to speak over charles**)

*-> Neutral rejection end

Action 9.2.3 – *ACCEPT*
- Your vision is very interesting, you got my support.
- I'm looking forward to the country that you will create.

*-> Acceptance end

Main 10.0
- The kingdom lackssh the neceshsary etiquette and eleganshe.
- I yearn for the day when all the citishenss of thiss kingdom…
- can reach even just a dusht of grassefulnesh as mysself.
- I would have hoped, of all people, you would at leasst undersstand me.
- I mussht have missundersstood your eleganshe for ssomething elshe.
- …
- It seems to me that I was right.

Action 10.1 – ACCEPT OFFER
- Are you really trying to improve the kingdom for the better?
- Of coursshe! I have a lot of planssh on how we can improve…
- Sso what do you ssay, do you want to transshform the kingdom…
- together with me?
- I'm looking forward to it!

*-> Acceptance end

Action 10.2 – KIND REJECTION
- I'm sorry, I already have too many troubles and being…
- involved in the Royal selection would only create more of them…
- Thanks for the lovely Tea but I'll have to humbly decline.

*-> Neutral rejection end

Action 10.3 – LASH OUT (Unlock if madam friendliness is 2 or lower)

- I disgust narcissistic and arrogant types like you.
- The elegance will only give an illusion of order.
- It all is just one big facade.
- By looking at the way you treat Charles,
- I don't think the nation will be handled by you any different.

*-> Mad rejection end

Action 10.4 - TALK CHARLES (UNLOCK if Charles friendliness is 6 or more)

- Before I answer, I have a question for Charles.
- I'm terribly ssorry for hissh behaviour please forgive him.
- CHARLESSH apologize to your honored guessht!
- I'm terribly sorry!
- I hope you can forgive me for my errors!

*-> Main 11

Action 10.4.1 - *DE-ESCALATE*

- No it's nothing of that kind. (blend over charles)

*-> Main 11

Main 11

- Charles,
- Y-Yes?
- How has your experience been while working for Lady madam?
- I-It has been a bliss. I am eternally grateful for…
- Lady Madam.. to let someone as incompetent as myself…
- serve her
- …
- …

*-> Neutral rejection end

Action 11.1 – *POACH CHARLES*

- Lady Madam is not treating you right..
- You deserve better!
- What nonsshense are you ssspouting! **blend over top**
- Come and work under me instead! **blend over top**
- You won't regret it!

*-> Mad rejection end **blend over top line**

Action 11.2 – *SUPPORT MADAM*

- If Charles says that it is bliss working with you, then…
- it would be a shame to let this opportunity go.
- You got my support!

*-> Acceptance end

Neutral rejection end

- Well thank you for coming and lisstening to my proposshal,
- It is unfortunate that I couldn’t get your ssupport…
- but it has been an delightful talk neverthelesssh…
- Now if you excuse me, I have ssome buisshnessh to attend to.
- Charlessh, pleasshe sshow our honored guessht the way out…
- when he hassh finished hissh tea.
- And treat him with outmossht resshpect.
- Yes Lady Madam, I will.

Mad rejection end

- I- I.
- Outrageoussh!
- I won’t forget this! Not even willing to listen!
- You will deeply \regret this when I become the ruler of the country.
- CHARLESSH — escort our guesht out of the premise.
- Ye- yes lady Madam, could you please follow me.

Acceptance end

- Wonderful, I knew you were a persshon with great visshion!
- This callssh for a cshelebration! CHARLESSh — Bring out more biscuits…
- Hurry up, sshtop sshlacking.
- Y- Yes Lady Madam, I’m coming with haste.
- I’m looking forward to getting to know you better!
- Likewise!