

Writing *Façade*: A Case Study in Procedural Authorship

Michael Mateas and Andrew Stern

1. INTRODUCTION

The essence of the computer as a representational medium is procedurality — the ability of the computer to engage in arbitrary mechanical processes to which observers can ascribe meaning. Computers do, of course, participate in the production of imagery, support communication between people via the mediation of long-distance signals, control electro-mechanical devices, and support the storage and interlinking of large quantities of human-readable data. Many tools are available that allow users to engage these various capacities of the computer, such as image manipulation or web page authoring, without requiring users to think procedurally. But it is precisely the computer's ability to morph into these special-purpose machines that highlights the computer's procedural nature. These special-purpose machines (e.g., tools) are made out of computational processes; the computer's ability to engage in arbitrary processes allows it to morph into arbitrary machines.

Taking full representational advantage of the computer thus requires procedurally literate authorship; that is, artists and writers who are able to think about and work within computational frameworks. By procedural literacy, we mean the ability to read and write processes, to engage in procedural representation and aesthetics, to understand the interplay between the culturally-embedded practices of human meaning-making and technically-mediated processes. Even for new media practitioners who don't themselves write much code, procedural literacy is necessary for successfully participating in interdisciplinary collaborative teams, and for understanding the space of possibility for digital works. Many authors find themselves engaged in some level of programming, especially for interactive work that, of necessity, requires conditional response to input, and thus the specification of a process. In the extreme case of developing new modes of computational expression, authors must be highly proficient in the use of general purpose programming languages, used to construct new languages and tools specialized for the new representational mode.

In this paper we provide a case study, using the interactive drama *Façade*, of this last case of procedural authorship. *Façade* represents a new mode of computational representation, *interactive drama*, combining the game-like pleasure of moment-by-moment interaction with believable characters and the story-like pleasure of participating in and influencing a long-term, well-formed dramatic progression. As procedural authors, we undertook several design-plus-programming tasks: deconstructing a dramatic narrative into a hierarchy of story and behavior pieces; designing an AI (artificial intelligence) architecture and a collection of special-purpose languages within the architecture, which respond to and integrate the player's moment-by-moment interactions to reconstruct a real-time dramatic performance from those pieces; and writing an engaging, compelling story within this new framework.

This paper makes a case for the importance of procedural authorship, describes the design goals of *Façade* and how these goals could only be met through a highly procedural approach to interactive narrative, and finally describes *Façade*'s architecture, content organization, and the experience of authoring within this framework.

2. PROCEDURALITY

Janet Murray has identified four essential properties of the computer as a representational medium: that computers are procedural, participatory, encyclopedic and spatial [17]. The procedural, of course, refers to the machinic nature of computers, that they embody complex causal processes, and in fact can be made to embody *any arbitrary* process. The participatory refers to the interactive nature of computers, that they can dynamically respond to outside signals, and be made to respond to those signals in a way that treats those signals as having the meaning ascribed to them by people (that is, non-arbitrary response). The encyclopedic refers to the vast storage capacity of digital computers, and their ability to organize, retrieve and index stored material. The spatial refers to the ability of digital computers to represent space, whether that is the physical space of virtual reality and games, or the abstract space of networks of information.

Various communities of practice tend to hold different properties as central. Here we provide a few examples of the privileging of various properties. For the Demoscene, a largely competition-oriented subculture with groups and individual artists competing against each other in technical and artistic excellence [18], procedurality is central; the aim is to procedurally generate as rich an audio-visual experience as possible using the minimum amount of stored content. The participatory is privileged in rhetorics of agency, control, and co-authorship, and has been adopted by communities as diverse as user-interface design, interactive art, and digital marketing. Database art privileges the encyclopedic, sometimes viewing all new media art practice as metaphorically related to the manipulation and resequencing of data stores. Spatiality is privileged by such diverse communities as virtual reality, game design, and hypertext.

While all of these properties play some role in various computational media, procedurality is the essential, defining property of computational media, without which the other properties could not exist.

Any participatory system requires the specification of potential action that is carried out in response to a stimulus. Capturing a space of potential action requires specifying a machine or process that can actualize the potential under different contingencies. In other words, participatory systems require procedurality. The converse is not true; there can be procedural systems that are not participatory, but rather execute a fixed process without accepting input. Many generative art systems, such as Aaron [10], exhibit procedurality without being participatory.

Encyclopedic systems are similarly dependent on procedurality. Without the ability to perform operations on data, to be able to access, re-sequence, search, modify, index and so forth, large data stores are useless. Without the procedural competencies of web search technologies, for instance, the web could literally not exist at its current scale. There would be no reason to create a new web page without the ability to relate the page to other, already published pages, and the ability for others to be able to find and view your page. Again, the converse is not true. Processes can create elaborate experiences from very small kernels; this capability is in fact the inspiration for the Demoscene.

The spatial is clearly a derivative property, a representational illusion actively maintained by a process. Graphical spatial representations make use of procedural models to compute and dynamically update the displayed space. Interactive spaces, which create the sense of space by supporting active navigation through the space, and which may not make use of 2D or 3D graphical representations at all, depend on the participatory, which in turn is dependent on procedurality.

The goal here is not simply to play a dominance game between the various representational properties of computers, but to avoid serious confusions and misunderstandings that can arise in new media theory and practice from misunderstanding the central importance of procedurality. Without a deep understanding of the relationship between what lies on and beneath the screen, scholars are unable to deeply read new media work, while practitioners, living in the prison-house of “art-friendly” tools, are unable to tap the true representational power of computation as a medium

Without an understanding of procedurality, of how code operates as an expressive medium, new media scholars are forced to treat the operation of the media artifacts they study as a black box, losing the crucial relationship between authorship, code, and audience reception. Code is a kind of writing; just as literary scholars wouldn’t dream of reading only translated glosses of work, never reading the full work in its original language, so new media scholars must read code, not just at the simple level of primitive operations and control flow, but at the level of the procedural rhetoric, aesthetics and poetics encoded in a work.

New media practitioners without procedural literacy are confined to producing those interactive systems that happen to be possible to produce within existing authoring tools. To date, such tools tend to have an encyclopedic orientation; in the absence of significant support for procedural authorship (i.e., programming), authorship consists of the gathering together of numerous media assets (video, sound, text, image, etc.), and the spatial and temporal composition of those assets within the procedural framework supported by the tool (e.g., linking). This approach fundamentally limits the size and complexity of new media artifacts. For interactive works, this problem is especially severe, as it forces the author to pre-specify and explicitly author responses to all possible interactive situations.

2.1 Procedurality and Content

To describe the relationship between computation and media assets, Chris Crawford introduced the term *process intensity* [4]. Process intensity is the “crunch per bit,” the ratio of computation to the size of the media assets being manipulated by the system. If a game (or any interactive software) primarily triggers media playback in response to interaction, it has low process intensity. The code is doing very little work — it’s essentially just shoveling bits from the hard drive or CD-ROM to the screen and speakers. As a game (or any interactive software) manipulates and combines media assets, its process intensity increases. Algorithmically generated images and sound that make no use of assets produced offline have maximum process intensity.

Process intensity directly enables richness of interactivity. As process intensity decreases, the author must produce a greater number of offline assets (e.g., pre-rendered chunks of text, animations or video) to respond to the different possible interactions. The number of offline assets required to maintain a given level of interactivity increases exponentially as process intensity decreases; therefore, in general, decreases in process intensity result in decreases in the richness of interactivity.

Though games have a relatively high process intensity within the space of new media artifacts, contemporary games are pushing against authoring limits caused by an over-reliance on non-procedural, static assets. Contemporary games such as Electronic Arts’ *The Lord of the Rings* franchise currently contain more media files than lines of code [19]. Even open-world games such as the *Grand Theft Auto* franchise, lauded for their simulated, procedural worlds, still use static assets for every vehicle, every type of person, every building, every weapon, and so forth.

Furthermore, developers at a recent Game Developers Conference voiced concern that next-generation console game hardware will only exacerbate this content crisis [20]. The requirement for ever-more detailed graphics to entice consumers to purchase next-generation consoles means that assets become more expensive to produce, requiring ever-larger teams, making games more expensive. Consumers want more gameplay, meaning larger games, thus requiring even more assets to be produced; this all results in a positive-feedback loop that is considered by many to be unsustainable.

Where insufficient procedurality is creating a crisis in the authoring of traditional games, it has prevented some long sought-after genres of interactive art and entertainment, such as the high-agency interactive story, from even getting off the ground. Bringing process intensive, AI-based techniques to the problem of interactive story was one of the fundamental research goals of our interactive drama *Façade*.

3. PROCEDURAL CONTENT IN THE INTERACTIVE DRAMA *FAÇADE*

3.1 A Case Study for Procedural Content

Motivated by our belief that the research into highly procedural authoring methods will enable yet-to-be-realized genres of interactive art and entertainment, we undertook the development of the interactive drama *Façade* [7]. The dream of interactive drama, perhaps best envisioned by the *Star Trek* Holodeck and first presented in an academic context by Brenda Laurel in *Computers as Theatre* [8], has players interacting with compelling, psychologically complex characters, and through these interactions having a real influence on a dynamically evolving storyline. Using a decade of prior research from the Carnegie Mellon Oz Project [2, 9] as a starting point and our belief that a fully-realized interactive drama had not yet been built, we embarked on a five-year effort to develop procedural authoring methods for believable characters, natural language conversation, and a dynamic storyline, integrated into a small but complete, playable experience. Publicly released in July 2005, *Façade* has been downloaded by over 150,000 players worldwide as of this writing, and has received widespread critical acclaim [6].



we want the player to experience *global agency*, that is, longer-term player influence on the overall story arc, over which topics get brought up, how the characters feel about the player over time, and how the story ends.

Enjoyable video games tend to be highly procedural in implementation, because among implementation methods, procedurality affords the greatest degree of dynamism and reactivity — features very satisfying to players. The best procedural video games excel at giving players high-agency experiences; that is, providing ample opportunities for the player to take action and receive immediate feedback. With *Façade* we wanted to create an interactive drama that provides the level of immediate, moment-by-moment agency, i.e., *local agency*, found in games. But unlike games,

Like contemporary games, *Façade* is set in a simulated world with real-time 3D animation and sound, and offers the player a first-person, continuous, direct-

Figure 1. Grace and Trip in *Façade*, viewed from the player's first-person perspective.

interaction interface, with unconstrained navigation and the ability to pick up and use objects. But like drama, particularly theatrical drama about personal relationships such as *Who's Afraid of Virginia Woolf?* [1], *Façade* uses unconstrained natural language and emotional gesture as a primary mode of expression for all characters, including the player. Rather than being about saving the world, fighting monsters or rescuing princesses, the story is about the emotional entanglements of human relationships, specifically about the dissolution of a marriage. There is unity of time and space — all action takes place in an apartment — and the overall event structure is modulated to align to a well-formed Aristotelian tension arc, i.e., inciting incident, rising tension, crisis, climax, and denouement, independent of the details of exactly what events occur in any one run-through of the experience.

Additionally, the story-level choices in *Façade* are intended to not feel like obvious branch points. We believe that when a player is faced with obvious choice points consisting of a small number of choices (e.g., being given a menu of three different possible things to say), it detracts from the sense of agency; the player feels railroaded into doing what the designer has dictated. Instead, in *Façade*, the story progression changes in response to many small actions performed by the player throughout the experience.

Section 4 of this paper describes *Façade*'s procedural content in detail, and how it achieves these design goals.

3.2 Hindrances Of Low- or Non-Procedural Content

Authors have faced a long-time conundrum when undertaking the construction of interactive stories: how can a story be structured to incorporate interaction, yet retain a satisfying, well-formed plot when experienced by the reader/player? Historically the designs of low- or non-procedural interactive stories have been forced to make a tradeoff between these two goals; the resulting “interactive story” may have a well-formed plot, but can only be minimally influenced by the reader/player, as seen in the linear narrative threads of most games and some text-adventure interactive fiction (IF).

Alternatively, the design tradeoff may be made in the other direction, resulting in interactive experiences that can vary significantly as a result of player action, but lack the degree of coherency, pacing and focus that are pleasurable in well-constructed stories. A non-procedural, encyclopedic design approach, in which the author creates a large number of static story pieces (assets) that are sequenced by a simple system, inevitably forces this design tradeoff. The author can choose to place minimal constraints on the ordering of story pieces, allowing the local sequencing of pieces to depend on the local player interaction. But then the sequences produced will lack the coherency of well-formed story arcs. Fragmented plots, or plots heavily diluted with unorganized or non-useful bits of action, are common in hypertext fiction as well as some IF, making them problematic to characterize as proper stories.

Within an encyclopedic design approach, the only way to increase interactivity is to author extraordinary amounts of content by brute force. This strategy has been borne out to be impractical; even the most successful *Choose Your Own Adventure* books or their digital equivalents, where the plot may vary significantly in response to reader's choices *and* be well-formed, necessarily offer an unsatisfyingly short series of infrequent, binary choices in order to avoid a combinatorial explosion of explicitly rendered (pre-written) plot directions. In such an approach, the limited and cumbersome nature of a non-procedural, encyclopedic approach is exposed.

The encyclopedic tradeoff between coherency and the combinatorial explosion seen at the plot level is mirrored at the more detailed level of character dialogue. The low-coherency, simple-process approach to dialogue is exemplified by chatterbots, in which lines of dialogue are sequenced from a large pool in response to each player interaction, making use of little to no context, and depending primarily on simple stimulus/response rules. The high-coherency *Choose Your Own Adventure*-approach to dialogue is exemplified by dialogue trees, in which an author must explicitly and statically represent discourse context by pre-specifying all possible paths through the dialogue, resulting in the same combinatorial explosions suffered by story graphs.

Based on such frustrating limitations in prior approaches to interactive story, local and global agency have commonly been seen as incompatible.

3.3 Procedural Story Design

Our solution in *Façade* to this long-time conundrum is to recast player interactions within a story in terms of abstract *social games*. Games, which are procedural by nature, achieve the high degree of event variability and player agency that we desire; the challenge becomes how to design and structure games that reflect the particular meanings we wish our story to exhibit, and how to *dramatically perform* the games as coherent, focused, well-paced narratives.

Further, to be compatible with the procedural, simulation-oriented nature of games, the granularity of immutable story content pieces must be made unusually small, on the order of individual and re-combinable facial expressions, gestures and lines of dialogue, rather than multi-sentence lexias of text or extended cutscenes. As described in detail in Section 4, *Façade*'s content pieces are organized into multiple, mixable hierarchical levels, sequenced by procedures written in multiple, mixable authoring languages.

At a high level, *Façade*'s abstract social games are organized around a numeric "score," such as the affinity between a character and the player. However, unlike traditional video games where there is a fairly direct connection between player interaction (e.g., pushing a button to fire a gun) and score state (e.g., a decrease in the health of a monster), *Façade*'s social games have several levels of abstraction separating atomic player interactions from changes in social "score." Instead of jumping over obstacles or firing a gun, in *Façade* players fire off a variety of *discourse acts* in natural language, such as praise, criticism, flirtation and provocation (see Table 1). While these discourse acts will generate immediate reactions from the characters, it may take story-context-specific patterns of discourse acts to influence the social game score. Furthermore, the score is not directly communicated to the player via numbers or sliders, but rather via enriched, theatrically dramatic performance.

As a friend invited over for drinks at a make-or-break moment in the collapsing marriage of the protagonists Grace and Trip, the player unwittingly becomes an antagonist of sorts, forced by Grace and Trip into playing psychological "head games" with them [3]. During the first part of the story, Grace and Trip interpret all of the player's discourse acts in terms of a zero-sum *affinity game* that determines whose side Trip and Grace currently believe the player to be on. Simultaneously, the *hot-button game* is occurring, in which the player can trigger incendiary topics such as sex or divorce, progressing through tiers to gain more character and backstory information, and if pushed too far on a topic, affinity reversals. The second part of the story is organized around the *therapy game*, where the player is (purposefully or not) potentially increasing each characters' degree of self-realization about their own

problems, represented internally as a series of counters. Additionally, the system keeps track of the overall story *tension level*, which is affected by player moves in the various social games. Every change in each game's state is performed by Grace and Trip in emotionally expressive, dramatic ways. On the whole, because their attitudes, levels of self-awareness, and overall tension are regularly progressing, the experience takes on the form and aesthetic of a loosely-plotted domestic drama.

As the granularity of the atomic pieces of story content (e.g., dialogue, emotion and gestural expression) becomes very small, and the procedures to sequence and combine them into a coherent narrative performance become primary to the realization of the experience for the player, the author's activity shifts from that of a writer of prose into a writer of procedures; that is, into becoming a programmer.

3.4 Richness Through Coherent Intermixing

To dramatically perform *Façade*'s social games as coherent, focused, well-paced narratives, an organizing principle is required that breaks away from the constraints of traditional branching narrative structures, to avoid the combinatorial explosion that occurs with complex causal event chains [5]. Our approach to this in *Façade* is twofold: first, we divide the narrative into *multiple fronts of progression*, often causally independent, only occasionally interdependent. Second, we build a variety of *narrative sequencers* to sequence these multiple narrative progressions. These procedural sequencers, described below, operate in parallel and can coherently intermix their performances with one another.

Façade's architecture and content structure are two sides of the same coin, and will be described in tandem; along the way we will describe how the coherent intermixing is achieved.

3.4.1 Architecture and Content Framework

The *Façade* system consists of several procedural subsystems that operate simultaneously and communicate with one another [11, 12, 13, 14, 15]. Each is briefly described below.

The dynamic, moment-by-moment performance of the characters Grace and Trip — how they perform their dialogue, how they express emotion, how they follow the player around and use objects — are written as a vast collection of *behaviors*, which are short reactive procedures representing numerous goals and sub-goals for the characters, arranged in a vast, hierarchical, dynamically-changing tree structure. These behaviors are written in a reactive-planning language called A Behavior Language (ABL), developed as part of the *Façade* project, that manages both parallel and sequential behavior interrelations such as sub-goal success and failure, priority, conflict, preconditions and context conditions.

The narrative sequencers for the social games are also written in ABL, taking advantage of ABL's ability to perform *meta-behaviors* that modify the runtime state of other behaviors. (More on this in the next section.)

The highest-level narrative sequencer, a subsystem called the *drama manager*, sequences dramatic *beats* according to specifications written in a custom drama management language. Beats in *Façade* are large groups of behaviors organized around a particular topic, described in the next section.

Another subsystem is a set of rules for understanding and interpreting natural language (NL) and gestural input from the player. These rules are written in a custom language implemented with Jess, a forward-chaining rule language. When the player enters dialogue, these NL rules interpret one or more meanings (the aforementioned *discourse acts*). A second set of rules called *reaction proposers* further interpret these discourse acts in context-specific ways, such as agreement, disagreement, alliance, or provocation, and send this interpretation to the behaviors and drama manager to react to.

agree, disagree, generalExclamation, positiveExclamation, negativeExclamation, express <emotion>, maybeUnsure, dontUnderstand, thank, apologize, referTo <character> | <object> | <topic> | <theme>, physicallyFavor <object>, praise, hugComfort, flirt, kiss, showConcern, howAreYou, areYouOkay, showSupport, pacify, criticize, oppose, greet, goodbye, getAttention, inappropriateObscene, explain <explainAdviceDescriptor>, advice <explainAdviceDescriptor>, explainRelationship <character1> <relationshipDescriptor> <character 2>, leaveApartment, leaveForKitchen, uncooperativeNotSpeaking, uncooperativeNotMoving, uncooperativeFidgety, systemDoesntUnderstand

Table 1. *Façade*'s discourse acts.

The final subsystem is a custom animation engine that performs character action, emotional expression and spoken dialogue by way of real-time non-photorealistic procedural rendering, as well as music and sound. The animation engine is driven by the ABL behaviors; the engine also senses information about the location and actions of each character for the behaviors to use.

3.4.2 Beats, Beat Goals and Beat Mix-ins

Façade's primary narrative sequencing occurs within a beat, inspired by the smallest unit of dramatic action in the theory of dramatic writing [16]. However *Façade* beats ended up being larger structures than the canonical beats of dramatic writing. In dramatic writing, a beat tends to consist of just a few lines of dialogue that convey a single narrative action/reaction pair¹. A *Façade* beat, however, is comprised of anywhere from 10 to 100 *joint dialogue behaviors* (JDBs), written in ABL. Each beat itself is a narrative sequencer, responsible for sequencing a subset of its JDBs in response to player interaction. Only one beat is active at any time. A JDB, *Façade*'s atomic unit of dramatic action (and closer to the canonical beat of dramatic writing), consists of a tightly coordinated, dramatic exchange of 1 to 5 lines of dialogue between Grace and Trip, typically lasting a few seconds. JDBs typically consist of 50 to 200 lines of ABL code. A beat's JDBs are organized around a common narrative goal, such as a brief conflict about a topic, like Grace's obsession with redecorating, or the revelation of an important secret, like Trip's attempt to force Grace to enjoy their second honeymoon in Italy. Each JDB is capable of changing one or more values of story state, such as the affinity game's value, or any of the therapy game's self-

¹ For example, in the scene in *Casablanca* where Rick confronts Ilsa about why she returned, the following exchange forms a single beat: RICK: "Why'd you come back? To tell me why you ran out on me at the railway station?" ILSA: "Yes."

revelation progression counters, or the overall story tension level. Within-beat narrative sequencers implement the affinity game; the topic of a beat is a particular instance of the affinity game.

Each beat can be viewed as a bag of procedural content, specifically JDBs, which are dynamically sequenced by the specific logic of each beat. The drama manager is in turn a bag of procedural content, specifically beats, which are dynamically sequenced by the general logic of the drama manager, as influenced by the preconditions, weights, priorities, etc. specified for each beat. The logic required to sequence individual lines of dialogue is more detailed and complex than can be easily described in the declarative annotations at the drama management level; this is precisely why our beats turned out to be larger than traditional beats of dramatic writing.²

There are two typical uses of JDBs within beats: as *beat goals* and *beat mix-ins*. A beat consists of a canonical sequence of narrative goals called beat goals. The typical canonical sequence consists of a transition-in goal that provides a narrative transition into the beat (e.g., bringing up a new topic, perhaps connecting it to the previous topic), several body goals that accomplish the beat (in affinity game beats, the body goals establish topic-specific conflicts between Grace and Trip that force the player to choose sides), a wait goal in which Grace and Trip wait for the player to respond to the head game established by the beat, and a default transition-out that transitions out of the beat in the event of no player interaction. In general, transition-out goals both reveal information and communicate how the player's action within the beat has changed the affinity dynamic.

A beat's canonical beat goal sequence captures how the beat would play out in the absence of interaction. In addition to the beat goals, there is a set of meta-behaviors, called *handlers*, which wait for specific interpretations of player dialogue (discourse acts), and modify the canonical sequence in response, typically using beat mix-ins. That is, the handler logic implements the custom narrative sequencer for the beat. Beat mix-in JDBs are beat-specific reactions used to respond to player actions and connect the interaction back to the canonical sequence. Handlers are responsible both for potentially adding, removing and re-ordering future beat goals, as well as interjecting beat mix-ins into the canonical sequence. By factoring the narrative sequencing logic and the beat goals in this way, we avoid having to manually unwind the sequencing logic into the beat goal JDBs themselves, thus avoiding the dialogue tree problem mentioned in Section 3.2.

For *Façade*, an experience that lasts ~20 minutes and requires several replays to see all of the content available (any one run-through performs at most 25% of the total content available), we authored ~2500 JDBs. Approximately 66% of those 2500 are in beat goals and beat mix-ins, organized into 27 distinct beats, of which ~15 are encountered by the player in any one run-through (see the drama management section further below).

² The detailed sequencing and coordination of individual lines of dialogue is more readily expressed in ABL than in the beat description language, and in fact changes enough from context to context within the drama, that a generic decision-making process for sequencing lines of dialogue is not feasible (at least, not without much deeper knowledge representation, deep reasoning about human social interaction, including common-sense reasoning, etc.). Thus, we push that detailed logic into the custom narrative sequencers, written in ABL, that live within each beat, leaving the drama manager to sequence larger blocks of narrative content whose interrelationships are simple enough that they can be managed by the more generic decision-making process operating at this level.

3.4.3 Global Mix-in Progressions

Another type of narrative sequencer, which operates in parallel to and can intermix with beat goals and beat mix-ins, are *global mix-ins*. (How coherent intermixing is achieved is described in a later section.) Each category of global mix-in has three tiers, progressively digging deeper into a topic; advancement of tiers is caused by player interaction, such as referring to the topic. Each tier in the progression is constructed from one or more JDBs, just like beat goals or beat mix-ins. They are focused on satellite topics such as marriage, divorce, sex, therapy; or about objects such as the furniture, drinks, their wedding photo, the brass bull, or the view; or as generic reactions to praise, criticism, flirtations, oppositions and the like. Additionally, there are a variety of generic deflection and recovery global mix-ins for responding to overly confusing or inappropriate input from the player. In total there are ~20 instances of this type of narrative sequencer in *Façade*, comprising about 33% of the total ~2500 JDBs.

3.4.4 Drama Management (Beat Sequencing)

The coarsest narrative sequencing in *Façade* occurs in the drama manager, or *beat sequencer*.

PlayerArrives, TripGreetPlayer, PlayerEntersTripGetsGrace, GraceGreetPlayer, ArgueOverRedecorating, ExplainDatingAnniversary, ArgueOverItalyVacation, FightOverFixingDrinks, PhoneCallFromParents, TransitionToTension2, GraceStormsToKitchen, PlayerFollowsGraceToKitchen, GraceReturnsFromKitchen, TripStormsToKitchen, PlayerFollowsTripToKitchen, TripReturnsFromKitchen, TripReenactsProposal, BlowupCrisis, PostCrisis, TherapyGame, RevelationsBuildup, Revelations, EndingNoRevelations, EndingSelfRevelationsOnly, EndingRelationshipRevelationsOnly, EndingBothNotFullySelfAware, EndingBothSelfAware

Table 2. The names of *Façade*'s 27 beats.

This lies dormant most of the time, only active when the current beat is finished or is aborted (by the beat's own decision, or by a global mix-in). It is at the beat sequencing level where causal dependence between major events is handled — that is, where high-level plot decisions are made.

In a beat sequencing language, the author annotates each beat with selection knowledge consisting of preconditions, weights, weight tests, priorities, priority tests, and story value effects — the overall tension level, in *Façade*'s case. Given a collection of beats represented in the beat language, such as the 27 listed in Table 2, the beat sequencer selects the next beat to be performed. The unused beat whose preconditions are satisfied and whose story tension effects most closely match the near-term trajectory of an author-specified story tension arc (in *Façade*, an Aristotelian tension arc) is the one chosen; weights and priorities also influence the decision. [13].

Subsequent sections on Coherent Intermixing, and Failures and Successes, further discuss beat sequencing.

3.4.5 Long-term Autonomous Mix-in Behaviors

Long-term autonomous behaviors, such as fixing drinks and sipping them over time, or compulsively playing with an advice ball toy, last longer than a 60-second beat or a 10-second global mix-in. While perhaps performing only a minor narrative function, occasionally mixing in a JDB into the current beat (comprising only 1% of *Façade*'s JDBs), they contribute a great deal to the appearance of intelligence in the characters, by having them perform extended, coherent series of low-level actions in the background over the course of many minutes, across several beat boundaries. By simultaneously performing completely autonomous behaviors and joint behaviors, *Façade* characters are a hybrid between the “one-mind” and “many-mind” extremes of approaches to agent coordination, becoming in effect “multi-mind” agents [11].

3.5 Strategies for Coherent Intermixing

Since global mix-ins for the hot-button game are sequenced among beat goals/mix-ins for the affinity game, which both operate in parallel with the drama manager that is occasionally progressing overall story tension, several strategies are needed to maintain coherency, both in terms of discourse management and narrative flow.

First, global mix-in progressions are written to be causally independent of any beats' narrative flow. For example, while quibbling about their second honeymoon in Italy, or arguing about what type of drinks Trip should serve (affinity game beats, chosen by the drama manager), it is safe to mix in dialogue about, for example, sex, or the wedding photo (hot-button game mix-ins, triggered by a player's reference to their topics). Each mix-in's dialogue is written and voice-acted as if they are slightly tangential topics that are being juttred into the flow of conversation (“Oh, that photo, yeah, it's really ...”).

At the discourse level, mechanisms exist for smoothly handling such interruptions. During a beat goal, such as Trip's reminiscing about the food in Italy, if a global mix-in is triggered, such as the player picking up (thereby referring to) the brass bull, a gift from Trip's lover, the current Italy beat goal will immediately stop mid-performance, and the brass bull global mix-in will begin performing, at whichever tier that hot-button game has already progressed to. At the time of interruption, if the Italy beat goal had not yet passed its *gist point*, which is an author-determined point in a beat goal's JDBs, it will need to be repeated when the global mix-in completes. Short alternate uninterruptible dialogue is authored for each beat goal for that purpose. Also, each beat goal has a *reestablish* JDB that gets performed if returning to the beat from a global mix-in (“So, I was going to say, about Italy...”). Mix-ins themselves can be interrupted by other mix-ins, but if so, are not repeated as beat goals are.

With only a few exceptions, the affinity game beats themselves are also designed to be causally independent of one another. For example, in terms of maintaining coherency, it does not matter which order Grace and Trip argue about Italy, their parents, redecorating, fixing drinks, or their dating anniversary. When beat sequencing, this allows the drama manager to prefer sequencing any beats related to past topics brought up by the player. Likewise, hot-button mix-ins can be safely triggered in any order, into almost any beat at any time.

However, great authorial effort was taken to make the *tone* of each beat goal/mix-in and global mix-in match each other during performance. Most JDBs are authored with 3 to 5 alternates for expressing its narrative content at different combinations of player affinity and tension level. These

include variations in word choice, voice acting, emotion, gesture, and appropriate variation of information revealed. By having the tone of hot-button global mix-ins and affinity game beat goals/mix-ins always match each other, players often perceive them as causally related, even though they are not. Additionally, for any one tone, most JDBs are authored with 2 to 4 dialogue alternates, equivalent in narrative functionality but helping create a sense of freshness and non-roboticness in the characters between run-throughs of the drama.

4. DETAILED EXAMPLE OF AUTHORIZING PROCEDURAL CONTENT IN *FAÇADE*

To make concrete our discussion of authoring narrative and dialogue within a procedural framework, we will describe the process of authoring a specific story beat of the interactive drama *Façade*. Authoring a *Façade* beat involves a combination of interaction design, dialogue writing and programming, summarized here.

4.1 Designing the Core Structure of a *Façade* Beat

Our example will be the beat “FightOverFixingDrinks” in which Trip and Grace argue over what kind of drink to make for the player, intended to reveal some of the underlying tension between them, and to further develop their characters. In the first half of the drama during which this beat can occur, the couple Grace and Trip, whose marriage has reached its breaking point, are trying their best to act like nothing is wrong. Specifically in this beat we’ll have Trip use fixing drinks as way to brag about how well-off and cultured he thinks they are; Grace, however, emboldened by the presence of the player, will counter Trip with an attempted attack on Trip about his materialism and faux-sophistication. Both Grace and Trip will challenge the player to take sides on these differences.

We will first lay out a relatively simple outline for the beat, to which we can add additional richness as we go. We designed a basic structure for this beat as follows, as a sequence of beat goals:

- Transition-in to the beat — Trip brings up the idea of drinks.
- Trip makes an initial suggestion, with bragging; Grace initially reacts to the brag. They wait for a few seconds for a player response, if any.
- Grace counters with her own suggestion based on what the player said, attacking Trip; Trip resists. They wait for a few seconds for another player response, if any.
- Transition-out of the beat — Trip and Grace each react to the player’s decision, and Trip begins making the drinks.

It is important that each beat goal described above be relatively short, e.g., no more than 10 seconds each, ideally 5 seconds or less. A small granule size for beat goals allows other beat goals to be intermixed more easily into this sequence (as described in Section 4.2, below). If a beat goal were longer than 10 seconds, we’d want to split it up into smaller multiple beat goals.

4.2 Reactivity Adds Richness

Next we'll describe the additional reactivity requirements for this beat, which will add further richness to the interaction. These requirements include:

- At any time during the beat, the player should be able to interrupt what Grace and Trip are saying and get an immediate response of some sort. Whatever dialogue was interrupted should be re-spoken afterwards in a believable way, as needed.
- At any time during the beat, the player should be able to bring up other topics or do actions that are not directly related to the topic of fixing drinks, and still get a response from Grace and Trip, as described earlier in Section 3.4.3. These global mix-ins include progressing responses to tangential topics such as divorce, sex, or therapy, or about objects such as the furniture, their wedding photo, or the brass bull, or generic reactions to praise, criticism, flirtations, oppositions and the like. After the response, Grace and Trip should return to progressing the original beat itself, in a coherent way.
- Any time *after* the beat, once in another beat, the player should be able to refer to what previously happened during this beat and get a response of some sort; we call this a post-beat mix-in.

To support the above reactivity requirements, we will add the following specific features to the beat's structure:

- *Gist points*: each beat goal needs to be annotated with a *gist point*, to know how far into a beat goal the player must have gotten to avoid needing to repeat it if interrupted to perform some other mix-in.
- *Repeat-dialogue*: Each beat goal needs dialogue variation used in case the beat goal needs to be repeated, because it got interrupted in order to perform a mix-in.
- *Reestablish-dialogue*: Each beat goal needs a prefatory line of dialogue that can reestablish its context, in case the previous beat goal was a global mix-in and the current beat goal is returning to what it was talking about. These often play as a prefix to repeat-dialogue above.
- *Local-deflect-dialogue*: Each beat goal needs a small set of local deflect dialogue, to be used in case the player interrupts the beat goal with a very generic utterance, for which there is no appropriate global mix-in. These are essentially local mix-ins.

4.3 Performance in a Variety of Contexts Adds Richness

In addition to the reactivity requirements described above, we want this beat to operate in a variety of contexts. For example, its specific dialogue, and perhaps its structure, should vary if the beat is performed early in the drama when the tension is still low, versus a bit further along when the tension has increased. (Once the tension has reached a very high level, as authors we've decided that Trip won't be in the mood to fix anyone a drink, and this beat won't be allowed to occur.)

Also, the beat should vary in specific dialogue, and perhaps structure, if the player has been siding with Grace, or with Trip, or stayed neutral, independent of tension level. In fact, if the player's affinity changes *during* the beat, the beat should use its varying dialogue/structure appropriately.

Finally, this beat, by its nature, can be performed a second time, if enough time has passed since the first time it was performed. That is, if the player wants Trip to make a second drink for her, that should be possible. There needs to be enough internal dialogue and structure variation to avoid unbelievably repeating the same dialogue a second time.

To support such context variety, we will add the following specific features to our beat's structure:

- Each beat goal will be written with dialogue variations for each combination of tension level (low or medium) and each player affinity value (neutral, siding-with-Grace, siding-with-Trip), for a total of $2 \times 3 = 6$ variations.
- When the beat is occurring at the second (medium) tension level, we will author alternate transition-out beat goals (endings) for the beat, in which Grace reveals aloud one of Trip's façade-shattering alcohol-related secrets, such as a secret dislike of the taste of liquor, his secret job in college as a lowly bartender, or how he regularly sneaks off to a working-class sports bar down the street. We will divvy these up among the tension/affinity structure variations.

Meeting the requirements listed in this and the previous section contribute to creating *agency* for the player, because they allow the player to cause this beat to happen when she wishes. They also contribute to dramatic *believability*, because it only makes sense that drinks could be requested to be fixed at any time, at least until the tension level of the drama becomes too great. Without supporting these requirements, the timing and structure of the discourse and drama overall can seem arbitrarily and unnaturally constrained, significantly reducing agency and believability; i.e., the aforementioned problems with the status quo of commercial and non-commercial interactive stories.

4.4 Alternate Dialogue Adds Richness

Ideally each line of dialogue has several variations; e.g., 3-5 alternates, all with the same dramatic meaning but with different phrasings and word choice. While only one alternate will be heard for any line of dialogue per performance, the player will have the opportunity to notice this variation the next time she plays *Façade* and experiences this beat again, or if this beat happens a second time in the same session.

4.5 Parallel Behavior Adds Richness

Critical for lifelikeness and dramatic believability, Grace and Trip are required to perform expressive, parallel behavior as part of their beat goals:

- As Grace and Trip speak their dialogue, they should emote their current mood through facial expression, gaze and gesture. The specific dialogue they are speaking during the beat will affect their mood, of course, but overall mood can also be affected by whatever other events happened before this beat, as well as by whatever mix-ins may occur during the beat. For example, if a global mix-in occurs about divorce during this beat, that may sour Trip's mood, even if he started

off somewhat chipper about fixing drinks. Additionally, while a character is speaking, all non-speaking characters should dynamically react to the speaking character. This is why the author must write *joint* dialogue behaviors for each character; behavior must still be written for the non-speaking characters that control how they react to the dialogue being spoken by the speaking character.

- As characters speak their dialogue, they should tend to follow the player to wherever she walks within the room. This means that, in general, the dialogue should be written to not depend on where the character is standing when it is spoken³.
- At almost any time during this beat, we could have Trip autonomously decide to walk behind the bar and begin preparing drinking glasses as he speaks, in anticipation of pouring drinks. Like alternate-dialogue variation, this timing variation will be noticed in subsequent performances of this beat in this session or next. This requires the beat’s dialogue to be written to be believable whether or not Trip is behind the bar.

4.6 Simplifications/Abstractions to Reduce Complexity

There are a few aspects of the above design that can be simplified and/or abstracted to reduce the complexity of its implementation, while still achieving a satisfying level of agency and believability for the player.

- *Simplify the mapping of player utterances/actions to meanings, reducing the number of story reactions to author.* Ideally we would create a distinct reaction (plus alternate dialogue) for each discourse act the player could express, for each distinct context in a beat. However, there are dozens of supported discourse acts (see Table 1), and potentially as many contexts within a beat as there are beat goals; e.g., anywhere from 5 to 10 per beat. The permutations would result in hundreds of reactions to author per beat. Instead, to make this tractable, we *grouped* related discourse acts together in context-specific ways. For example, if Trip suggests a martini and is hoping for agreement from the player, several similar discourse acts can be grouped together to be interpreted as “agreement with Trip” in this context: agree (“yes”), positiveExclamation (“sweet!”), thank (“thanks”), express happy (“that makes me happy”), or a hug gesture. Though this requires authoring custom mappings per discourse context, it is less work than authoring dozens of individual reactions within every context⁴.

³ There are beats whose dialogue does depend on being performed in a specific location in the room; for example, the dialogue in *ArgueOverItalyVacation* requires Trip to stand near the Italy photo next to the bar and gesture towards it (or from behind the bar, as a special case, since the photo happens to be near the bar). The *FightOverFixingDrinks* beat, however, is one of the more common beat types that should be performable anywhere in the room.

⁴ Generally, each beat defines a discourse context, though there can be multiple distinct contexts within a single beat. The smallest discourse contexts are associated with individual beat goals, though beats may have sub-beat contexts that span several beat goals. Because we can’t always group the same discourse acts together, in general each beat will need custom mappings from discourse acts to beat-specific meanings. For example, though “thank” and “positiveExclamation” both have the beat-specific

- *Reduce causal dependencies.* In Section 4.2 above we laid out the design goal of allowing tangential topic reactions to mix in at any time (a.k.a. global mix-ins, described in Section 3). For example, after Trip suggests a martini, if the player mentions divorce, Trip needs to respond about divorce, and hopefully return to his martini suggestion afterwards. But couldn't the mention of divorce, or anything else, change the situation enough that it doesn't make sense to continue suggesting martinis, or whatever was being talked about beforehand? To keep this tractable, we try to design the narrative and to write the specific dialogue to reduce such causal dependencies. Trip's dialogue responding to the topic of divorce, while subtly revealing some hidden tension or feeling about it, has him trying to sweep it under the rug, allowing him to believably return to what he was talking about. When re-suggesting martinis afterwards, Trip's mood may darken a bit, altering his facial expressions and body language from that point forward, but not necessarily requiring the FightOverFixingDrinks to alter its structure significantly. This strategy of course results in reducing global agency — although the player did cause an immediate local response (a reaction to the player's mention of divorce), i.e., local agency, she causes fewer longer-term narrative effects (e.g., significantly changing the way drinks are discussed from that point onward). As authors we try to make up for that reduction in agency by *delaying* the narrative effect of having brought up divorce, responding to it later in the drama when it's easier to do so; e.g., in a beat (such as the BlowupCrisis beat) that explicitly recounts the provocative things the player said earlier.
- *Collapse contexts together when possible.* In Section 4.3 above we laid the design goal that each beat goal will be written with dialogue variations for each combination of tension level (low or medium) and each player affinity value (neutral, siding-with-Grace, siding-with-Trip), for a total of $2 \times 3 = 6$ variations. However some of these contexts are similar enough that they can be collapsed together. Specifically, in the case of a beat about Trip suggesting drinks to the player, as authors we could imagine that Trip would act with similar levels of braggadocio if he has affinity with the player or if the affinity is neutral, while acting differently if Grace has affinity with the player. Furthermore, as authors we could decide that once the tension has increased to a “medium” level, it makes no sense for player affinity to be neutral; if the player is still neutral when the tension raises to medium, we will force player affinity towards Trip or Grace. Each of these simplifications removes a context from the list, reducing the total to 4, thereby reducing the authoring burden for FightOverFixingDrinks by 33%.
- *Write the dialogue to allow for brief moments of uninterruptibility,* reducing the need for repeat-dialogue in case of interruption. As described in Section 4.2, each beat goal should have dialogue variation used in case the beat goal was interrupted by a mix-in and needs to be repeated. However, we can eliminate the need for repeat-dialogue for a beat goal if we can write the beat goal's dialogue to quickly communicate the gist of its meaning in its first few seconds, and annotate those first few seconds as *uninterruptible*. That is, if the player speaks during the first few seconds of such a beat goal, Grace and Trip's response is delayed until the beat goal's gist point is reached — a delay in reaction of a few seconds, which is just barely acceptable for believability. If the gist of the beat goal's meaning is communicated in those few seconds, we can interrupt the beat goal in order to perform a mix-in response to the interruption, and not bother repeating the interrupted beat goal later. This requires writing dialogue such that the minimum

meaning of agreement in the FightOverFixingDrinks beat, they may have distinct and different beat-specific meanings during other contexts in the drama.

amount of content required for the beat’s narrative progression to make sense is communicated close to the beginning of the beat goal, with the rest of the dialogue within the beat goal adding richness, color and additional detail to the basic content. As a general rule, the author must avoid long, complex lines of dialogue, instead breaking dialogue down into multiple lines that can be interrupted at line boundaries at a minimum (a fully interruptible line can of course be interrupted anywhere).

4.7 Authoring of a Beat Goal

Now that our design is in place, we are ready to author our beat goals. In the interest of space, we will only show the details of two beat goals for the `FightOverFixingDrinks` beat — the second and third beat goals listed in our core structure from Section 4.1, here given the names “`TripSuggest`” and “`GraceCounterSuggest`”:

- *TripSuggest*: Trip makes an initial suggestion, with bragging; Grace initially reacts to the brag. They wait a few seconds for a player response, if any. (Grace and Trip’s response to the player happens in the next beat goal, `GraceCounterSuggest`.)
- *GraceCounterSuggest*: Trip responds to the player, and Grace counters with her own suggestion based on what the player said, attacking Trip; Trip resists. They wait a few seconds for another player response, if any. (Grace and Trip’s response to the player happens in the next beat goal.)

We will write the dialogue in phases, starting off simple with just “`TripSuggest`,” and adding richness as we go. In each phase of the authoring, ***bold-italicized text*** will denote changes from the previous phase. In the interest of space, we will show pseudocode between angle brackets, not actual ABL behavior code. “T:” and “G:” denote dialogue spoken by Trip and Grace, respectively. Where the word “Player” appears in the dialogue, the player’s actual name is substituted, for example, “Brenda.”

4.7.1 Scaffolding

Here are some basic lines of dialogue for “`TripSuggest`” that can serve as scaffolding for the authoring process.

“`TripSuggest`”

T: (cheery) What would you like?

T: (cheery) How about a martini?

T: (bragging) I’m a real expert at fixing these, at least that’s what everybody tells me.

G: (a bit annoyed) Oh God, Trip, please ... let’s not go overboard with the drink preparation.

4.7.2 Uninterruptibility, Gist Point and Reestablish-Dialogue

As described in Section 4.6, we will set this beat goal to be uninterruptible at first, then set it to be interruptible and set its gist point a few seconds later. Also, we will prefix a line of reestablish-dialogue,

to be played if the context of the beat needs to be reestablished because of an interruption by a global mix-in.

“TripSuggest”

<set uninterruptible>

<if reestablish> T: (cheery) So! Drinks!

T: (cheery) What would you like?

T: (cheery) How about a martini?

<set interruptible>

<set gist point>

T: (bragging) I’m a real expert at fixing these, at least that’s what everybody tells me.

G: (a bit annoyed) Oh God, Trip, please ... let’s not go overboard with the drink preparation.

This means that if the player speaks early on in the beat goal, Trip won’t stop speaking until he’s done saying, “How about a martini?” Then the reaction will occur (whatever it is), and the last two lines will go unheard.

4.7.3 Custom Reactivity

The “TripSuggest” beat goal we are authoring is the second beat goal in this beat; what if earlier, during the first beat goal, the transition-in, the player requested a specific drink; e.g., said “I’d like a beer”? We should have dialogue variation in “TripSuggest” to react to that.

“TripSuggest”

<set uninterruptible>

<if reestablish> T: (cheery) So! Drinks!

<if nothing suggested so far>

T: (cheery) What would you like?

T: (cheery) How about a martini?

<set interruptible>

<set gist point>

T: (bragging) I’m a real expert at fixing these, at least that’s what everybody tells me.

<if a fancy drink was just requested>

T: (excited) Sure, that sounds great, I’ll have one too!

<if a boring drink was just requested>

T: (a bit down) Oh, but let’s enjoy ourselves tonight!

G: (a bit annoyed) Oh God, Trip, please ... let’s not go overboard with the drink preparation.

4.7.4 Map Player Utterances/Actions to Few Reactions

As described in Section 4.6, for reacting to the player in our “TripSuggest” beat goal, we will group similar discourse acts together, and map them to a small set of reactions. Here we map many of the discourse acts in Table 1 to just five reaction types:

- *AgreeTrip*: agree (“yes”), positiveExclamation (“sweet!”), thank (“thanks”), express happy (“that makes me happy”), hug.
- *DisagreeTrip*: disagree (“nah”), negativeExclamation (“lame”), express angry (“I’m pissed”), express sad (“I’m suddenly bummed out”).
- *SpecificFancyRequest*: referTo <fancyDrink> (“how about a cosmo?”, or “got any Scotch in there?”).
- *SpecificBoringRequest*: referTo <nonFancyDrink> (“just water for me,” or “Coors Lite, if you have it”).
- *NonAnswer*: timeout (many seconds of silence from the player), referTo drink (“a drink sounds good”), express laugh (“ha ha”), maybe (“I guess so”), dontUnderstand (“I don’t know what a martini is”), apologize (“sorry”), agree etc. towards Grace (“sure Grace,” “thanks Grace,” etc.).

Note that the remaining discourse acts from Table 1 not handled by this local mapping will still be handled by the global context⁵.

The reaction types listed above are implemented as dialogue variations in the beginning of the next beat goal, “GraceCounterSuggest”:

“GraceCounterSuggest”

<set uninterruptible>

<if reestablish> T: About those drinks ...

<if AgreeTrip> T: (excited) Good choice!

<if DisagreeTrip> T: (dismayed) What? Oh, I thought you’d love that ... !

<if SpecificFancyRequest> T: (excited) Oh yeah, that’s even better!

<if SpecificBoringRequest> T: (let down) Oh, but let’s celebrate tonight ... !

<if NonAnswer> T: (nervous) Uh, well, you know what, I’m just going to make you a martini.

<if current drink suggestion is fancy> G: No no, Player, maybe you’d like some juice, or a mineral water?

<if current drink suggestion is not fancy, e.g., a beer>

G: Trip, we don’t all share your infatuation with mixed drinks.

G: Player, you’d prefer just a beer, right?

<set interruptible>

<set gist point>

⁵ For example, if the player refers to sex during TripSuggest, since none of the TripSuggest-specific mappings handle references to sex, this context will suggest no reactions. The global context on the other hand will suggest a global mix-in reaction, which, since no more-specific context has a suggestion, will be selected as the reaction to perform. The global mix-in will be inserted between TripSuggest and GraceCounterSuggest; GraceCounterSuggest will then perform prefaced with its reestablish line. (Note that some other context in this or another beat might actually have a beat-specific response to a reference to sex, during which a global mix-in about sex would not be chosen to occur.)

T: (dismayed, under breath) Oh come on ...

4.7.5 Physical Performance

So far, our detailed beat goal authoring example has focused on the authoring of dialogue and dialogue logic. In addition, as mentioned in Section 4.5, the author needs to specify physical performance. This includes deciding where the characters should stand in relation to the player's current position (staging), how close each character should be to the player (often determined by affinity), changes in mood (influences facial expression and body stance), any gestures the characters should perform and how they are coordinated with the dialogue, base facial expression and momentary expressions (shock, surprise, etc.), and so forth. Besides participating in the dialogue logic, each JDB specifies procedural direction for how the character should perform its specific lines. Below is some example ABL code for a single JDB, in this case the JDB for the "TripSuggest" case where the player has not made any specific fancy drink request.

```
// ## if no specific fancy drink request (but includes if we had gotten a specific non fancy request)
joint parallel behavior bFAskDrinkT1NTPA_TripSuggest_BodyStuff() { teammembers Trip Grace;
  precondition { StoryMemory (BeatFAskDrinkT1WME bGotFancySpecificRequest == false
  curDrinkIdea :: drink) }

  with (ignore_failure, property isStagingGoal true)
    spawngoal StagingConverse(40, player, eConverseType_offCenterShared,
eWalkType_normal);
  with (persistent, team_effect_only) subgoal TryToKeepFacingSprite(0, player);
  with (priority_modifier 1, ignore_failure, team_effect_only)
    subgoal SetPerformanceInfo(40, 2, eHeadEmphType_nodStrong,
eArmsEmphType_atSide,
    startWith, -1, eGazeType_normal, player, eFEBase_serious);
  with (priority_modifier 2) subgoal SetMood(eMood_happy, eMoodStrength_barely, -1, 0);
  subgoal bFAskDrinkT1NTPA_TripSuggest_BodyStuff_NoSpReq_seq(drink);
  subgoal SetLetBeatGoalFinishFlag(true);
}
sequential behavior bFAskDrinkT1NTPA_TripSuggest_BodyStuff_NoSpReq_seq(int drink) {
  int doOptionalPreface;

  // optional preface
  mental_act { doOptionalPreface = randGen.nextInt(2); }
  with (ignore_failure) subgoal
bFAskDrinkT1NTPA_TripSuggest_BodyStuff_NoSpReq_dial1(doOptionalPreface);

  // possible second preface
  with (ignore_failure) subgoal
bFAskDrinkT1NTPA_TripSuggest_BodyStuff_NoSpReq_seq2(drink);
```

```

// the drink suggestion
with (ignore_failure) subgoal DoFullExpressionBase(70, eFEBase_pleasant);
with (ignore_failure) subgoal
bFAskDrinkT1NTPA_TripSuggest_BodyStuff_NoSpReq_dia3(drink);
}

```

The purpose of showing a code snippet here is not to go through the code in minute detail, but rather to point out a few features of the ABL code for JDBs:

- The first thing to note is that *it is* code; it is not some static data structure or cut scene, but is rather a dynamic little machine that knows how to perform these particular lines, can perform them anywhere in the room, even as the player walks around, can perform them even if the player is engaged in other long-term physical behaviors (e.g., Trip walking around with his advice ball) and thus might require substituting or suppressing physical movements. This is not a cut-scene or statically pre-scripted performance, but is rather a behavior that dynamically adjusts the performance.
- The joint parallel behavior (this one is for Trip) automatically synchronizes with a paired behavior in Grace, allowing them to tightly coordinate their performance, even as they each simultaneously engage in parallel, unsynchronized behavior. (Grace’s side of this joint behavior is not shown here.)
- With the parallel behavior the author is specifying a bunch of action that should happen at the same time, in this case that the character should initiate staging to the center of the player’s current view and share this position with another character, should try to keep facing the player as the player moves around, should perform their lines using strong head nods and arms-at-side gestures for dialogue emphasis, should be in a barely happy mood (this will combine with a serious base facial expression), and that they should perform a certain sequence of lines that start out uninterruptible (uninterruptibility will be turned off when the gist point is hit — this occurs in the details of dialogue behaviors that are not shown here).

At the beat-goal level, authoring for *Façade* combines being both a writer and a director, where both the dialogue logic and performance details are procedurally expressed.

4.7.6 Dialogue Variation for Tension Level, Player Affinity, and Alternate-Dialogue

To finish off our authoring example, we need to fully list the dialogue of the remaining permutations of the two tension levels (low, medium) and three player affinities (neutral, Trip, Grace) for the “TripSuggest” and “GraceCounterSuggest” beat goals, including alternate-dialogue variation within each. As described in Section 4.6, the neutral and Trip affinities have been combined into one, thereby reducing the total number permutations for this beat from $2 \times 3 = 6$, down to $2 \times 2 = 4$: TensionLow-AffinityNeutralTrip, TensionLow-AffinityGrace, TensionMedium-AffinityNeutralTrip, and TensionMedium-AffinityGrace. See the accompanying Extended Dialogue section for this listing.

5. EVALUATING *FAÇADE*

In this section we attempt to characterize the resulting degree of agency achieved in *Façade*, as well as failures and successes in terms of design, interface and system architecture.

5.1 Characterizing Agency

Creating player agency was a primary design goal for *Façade*, afforded by our approach of authoring highly procedural content.

5.1.1 Local Agency

When the player's actions cause immediate, context-specific, meaningful reactions from the system, we call this *local agency*. Furthermore, the greater the range of actions the player can take — i.e., the more expressive the interface — then the richer the local agency (again, if the responses are meaningful).

Façade offers players a continuous, open-ended natural language interface, as well as physical actions and gestures such as navigation, picking up objects, hugging and kissing. The millions of potential player inputs are mapped, using hundreds of the aforementioned NL rules, into one or more of ~30 parameterized discourse acts (DAs) such as praise, exclamation, topic references, and explanations; a second set of rules called reaction proposers interpret these DAs in context-specific ways, such as agreement, disagreement, alliance, or provocation.

Ideally there would be immediate, meaningful, context-specific responses available at all times for all DAs. In the actual implementation of *Façade*, in our estimation this ideal is reached ~25% of the time, where the player has a satisfying degree of real-time control over Grace and Trip's emotional state, affinity to the player, which topic is being debated, what information is being revealed, and the current tension level. But more often, ~40% of the time, only a partial ideal is reached: the mapping/interpretation from DA to reaction is coarser, the responses are more generic and/or not as immediate. Furthermore, ~25% of the time even shallower reactivity occurs, and ~10% of the time there is little or no reactivity. These varying levels of local agency are sometimes grouped together in temporal clusters, but also have the potential to shift on a moment-by-moment basis.

There are two main reasons for these varying levels of local agency. First, from a design perspective, at certain points in the overall experience it becomes necessary to funnel the potential directions of the narrative in authorially preferred directions, to ensure dramatic pacing and progress. Second, and more often the case, a lack of local agency is due to limitations in how much narrative content was authored (see the Failures section, below).

5.1.2 Global Agency

The player has *global agency* when the global shape of the experience is determined by player action. In *Façade* this would mean that the final ending of the story, and the particulars of the narrative arc that lead to that ending, are determined in a smooth and continuous fashion by what the player does, and that at the end of the experience the player can understand how her actions led to this storyline.

Façade attempts to achieve global agency in a few ways. First, beat sequencing (i.e., high level plot) can be influenced by what topics the player refers to; the sequencing can vary within the number of allowed permutations of the beats' preconditions and tension-arc-matching requirements. Even with only 27 beats in the system, technically there are thousands of different beat sequences possible; however, since most beats are causally independent, the number of *meaningfully different* beat sequences are few.

More significant than variations of beat sequences (“what” happened) are variations within beats and global mix-in progressions (“how” it happened). A variety of patterns and dynamics are possible within the affinity, hot-button and therapy games over the course of the experience; in fact these patterns are monitored by the system and remarked upon in dramatic recapitulations in the BlowupCrisis beat halfway through the drama, and in the RevelationsBuildup beat at the climax of the drama. A calculus of the final “scores” of the various social games is used to determine which of five ending beats gets sequenced, ranging from either Grace or Trip revealing one or more big hidden secrets and then deciding to break up and leave, or both of them too afraid to do anything, or both of them realizing so much about themselves and each other that they decide to stay together.

5.2 Failures and Successes of *Façade*

In this section we attempt to evaluate our results in creating the interactive drama *Façade*, whose design goals were strongly shaped by our procedural content-centric approach to implementation.

5.2.1 Agency

During the production of *Façade*, within our “limited” authoring effort (beyond the building of the architecture, *Façade* required ~3 person-years of just authoring, which is more than a typical art/research project but far less than a typical game industry project) we made the tradeoff to support a significant degree of local agency, which came at the expense of not supporting as much global agency. Combined with the reality that the time required to design and author JDBs is substantial, only 27 beats were created in the end, resulting in far lower global agency than we initially hoped for. As a result, we feel we did not take full advantage of the power of the drama manager’s capabilities.

Furthermore, because the specification of each joint dialogue behavior — spoken dialogue, staging directions, emotion and gesture performance — requires a great deal of authoring and is not automatically generated by higher-level behaviors or authoring tools, we are limited to the permutations of hand-authored, intermixable content. *Façade* is not generating sentences themselves — although it is generating sequences.

5.2.2 Feedback

A major challenge we encountered, at which we believe *Façade* falls short, is in always clearly communicating the state of the social games to the player. With traditional games, it is straightforward to tell players the game state: display a numeric score, or show the character physically at a higher platform, or display the current arrangement of game pieces. But when the “game” is ostensibly happening inside the characters’ heads, and if we intend to maintain a theatrical, performative aesthetic (and not display internal feelings via stats and slider bars, *a la The Sims*), it becomes a significant challenge. In our estimation *Façade* succeeds better at communicating the state of the simpler affinity and hot-button games than the more complex therapy game.

5.2.3 Interface

Another major challenge was managing the player’s expectations, raised by the existence of an open-ended natural language interface. We anticipated natural language understanding failures, which in

informal evaluations of *Façade* to date, occur ~30% of the time on average. This tradeoff was intentional, since we wanted to better understand the new pleasures that natural language can offer when it *succeeds*, which in *Façade* we found occurs ~70% of the time, either partially or fully.

5.2.4 System Architecture

In our estimation, a success of *Façade* is the integration of the beat goal/mix-in, global mix-in and drama manager narrative sequencers, with an expressive natural language interface, context-specific natural language processing, and expressive real-time rendered character animation. We feel the overall effect makes some progress towards our original design goals of creating a sense of the immediacy, presence, and aliveness in the characters required for theatrical drama.

As is evident from our authoring example, there is still significant effort in authoring an interactive drama within our architecture. Our architecture now makes authoring interactive drama possible, but not necessarily easy (it was extremely cumbersome or impossible using traditional finite state machine, dialogue tree, and story graph approaches).

It is unclear if there will ever be non-programming tools for authoring interactive drama; we believe it fundamentally requires procedural authorship. However, the idioms we have developed for structuring dialogue and using ABL within *Façade* can serve as a specification for a higher-level tool that facilitates authoring *Façade*-like experience. Even while authoring *Façade*, we were able to capture the general beat structure as an ABL code template that we could copy and modify for creating new beats. An obvious next step is to push these idioms as first-class language structures into ABL, or perhaps into a higher-level language that sits on top of ABL.

In general, our approach for architecting interactive drama systems is not to build a one-size-fits-all generic tool that tries to hide the fundamentally procedural nature of the medium, but rather to write languages for procedural authorship, build new experiences with those languages, and push the idioms and lessons learned from authoring prior experiences into first-class constructs in future languages.

5.2.5 Design

Certain aspects of our drama's design help make *Façade* a pleasurable interactive experience, while others hurt. It helps to have *two* tightly-coordinated non-player characters who can believably keep dramatic action happening, in the event that the player stops interacting or acts uncooperatively. In fact, the fast pace of Grace and Trip's dialogue performance discourages lengthy natural language inputs from the player. By design, Grace and Trip are *self-absorbed*, allowing them to occasionally believably ignore unrecognized or unhandleable player actions. Creating a loose, sparsely plotted story afforded greater local agency, but provided fewer opportunities for global agency. However, the richness of content variation, and at least a moderate degree of global agency achieved, does encourage replay.

The huge domain of the drama, a marriage falling apart, arguably hurt the success of the overall experience, in that it overly raised players' expectations of the characters' intelligence, psychological complexity, and language competence. As expected, the system cannot understand, nor has authored reactions for, many reasonable player utterances. The large domain often requires mapping millions of potential surface texts to just a few discourse acts, which can feel muddy or overly coarse to the player.

Also, continuous real-time interaction, versus discrete (turn-taking) and/or non-real-time interaction, added a great deal of additional complexity and authoring burden.

6. CONCLUSION

In this paper we have argued that procedural authorship is required to take full advantage of the representational power of the computer as an expressive medium. Procedurality is an underlying support for all modes of digital authorship; while procedural literacy is not required to create digital work, new media practitioners without procedural literacy are confined to producing those interactive works that happen to be possible to produce within existing authoring tools. We made a case for the importance of procedural authorship, describing the design goals of a case study, the interactive drama *Façade*, and how these goals could only be met through a highly procedural approach to interactive narrative. Based on our experience both architecting and authoring *Façade*, we have found that procedural authorship is essential for enabling yet-to-be-realized genres of interactive art and entertainment.

7. REFERENCES

- [1] Albee, Edward (1962). *Who's Afraid of Virginia Woolf?* Signet.
- [2] Bates, Joseph (1992). "Virtual Reality, Art, and Entertainment." *Presence: The Journal of Teleoperators and Virtual Environments* 1 (1992): 133-138.
- [3] Berne, Eric (1964). *Games People Play*. New York: Grove Press.
- [4] Crawford, Chris (1987). "Process Intensity". *The Journal of Computer Game Development*, 2 (1987): http://www.erasmatazz.com/library/JCGD_Volume_1/Process_Intensity.html.
- [5] Crawford, Chris (1989). "Indirection". *The Journal of Computer Game Design*, 3 (1989): http://www.erasmatazz.com/library/JCGD_Volume_3/Indirection.html.
- [6] Grand Text Auto (2005). "Finally, the Curtain Opens on *Façade*". <http://grandtextauto.gatech.edu/2005/07/05/facade-is-released/>
- [7] *Façade, a one-act interactive drama*. Michael Mateas and Andrew Stern; Procedural Arts. 2005. <http://www.interactivestory.net>
- [8] Laurel, Brenda (1991). *Computers as Theatre*. Reading, MA: Addison-Wesley.
- [9] Loyall, Bryan (1997). *Believable Agents*. Ph.D. thesis, Tech report CMU-CS-97-123, Carnegie Mellon University.
- [10] McCorduck, Pamela (1991). *Aaron's Code: Meta-art, Artificial Intelligence, and the Work of Harold Cohen*. New York, NY: W. H. Freeman and Co.
- [11] Mateas, Michael and Stern, Andrew (2004). "A Behavior Language: Joint Action and Behavioral Idioms." In *Life-like Characters: Tools, Affective Functions and Applications* edited by Helmut Predinger, and Mitsuru Ishioka. Springer.
- [12] Mateas, Michael and Stern, Andrew (2004). "Natural Language Understanding in *Façade*: Surface Text Processing." *Proceedings of Technologies for Interactive Digital Storytelling and Entertainment (TIDSE '04)*. Darmstadt, Germany.

- [13] Mateas, Michael and Stern, Andrew (2003). “Integrating plot, character and natural language processing in the interactive drama *Façade*.” Proceedings of the *1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment (TIDSE '03)*. Darmstadt, Germany.
- [14] Mateas, Michael and Stern, Andrew (2003). “*Façade*: an experiment in building a fully-realized interactive drama”. *Game Developers Conference (GDC '03)*, San Jose, CA, USA.
- [15] Mateas, Michael. and Stern, Andrew (2000). “Towards Integrating Plot and Character for Interactive Drama.” *Working notes of the Social Intelligent Agents: The Human in the Loop Symposium*. AAAI Fall Symposium Series. Menlo Park, CA.: AAAI Press.
- [16] McKee, Robert (1997). *Story: Substance, Structure, Style, and the Principles of Screenwriting*. New York: HarperCollins.
- [17] Murray, Janet (1998). *Hamlet on the Holodeck*. Cambridge, MA: MIT Press.
- [18] Wikipedia (2005). <http://en.wikipedia.org/wiki/Demoscene>
- [19] Grand Text Auto (2005). “Fever-addled impressions of GDC.”
<http://grandtextauto.gatech.edu/2005/03/23/fever-addled-impressions-of-gdc/>
- [20] Wonderland (2005). “Burn Down the House.”
http://crystaltips.typepad.com/wonderland/2005/03/burn_the_house_.html

8. EXTENDED DIALOGUE LISTING

“TripSuggest” for TensionLow and PlayerAffinityNeutral/Trip

<set uninteruptible>

<if reestablish>
T: So! Drinks!

<if nothing suggested so far>
T: What would you like?
T: So, what's your poison?

T: How about something fun,
T: Let's have something fun,
T: We should have something fun,

T: How does a martini sound?
T: How about a martini?
T: like a cosmopolitan?
T: like margaritas?
T: like sangria?
T: I've got the perfect bottle of cabernet I've been saving for just such an occasion.
T: Can I interest you in a single malt Scotch? It's primo.

<set interruptible>

<if a fancy drink was just requested>
T: Martinis... Great idea! Classic!
T: Ah, martinis... [Player], you've always had good taste.
T: Fabulous suggestion. I'll have the same.

T: Sure, that sounds great, I'll have one too!

<if a boring drink was just requested>

T: (wanting more) Oh, but we can do better than that...!

T: (wanting more) Oh, but let's enjoy ourselves tonight...!

T: (wanting more) Oh, but let's celebrate tonight...!

T: (wanting more) Oh, but let's live it up tonight...!

<brag>

T: I'm a real expert at fixing these, at least that's what everybody tells me.

T: Everybody tells me I fix the best drinks, so I'm sure you're gonna love this.

T: I just got a hold of this rare imported Icelandic vermouth I want you to try!

T: It's what we drink at these high-class poker games I go to with the execs at work.

T: Grace's dad taught me how to make these, it's a really classy drink.

T: The guy I play squash with introduced me to this drink, it's really amazing.

T: We went wine tasting last year in Napa with Grace's parents and discovered this stuff, it's exquisite.

T: I served these at our last party, they were a smash.

T: It's the latest thing, you'll love it.

<Grace reaction to brag>

G: (a bit muted) Oh God, Trip, please... let's not go overboard with the drink preparation.

G: (a bit muted) (slightly annoyed) uhh... you and your 'high class' drinks...

G: (a bit muted) We don't need to make a big production out of this, Trip.

G: (a bit muted) Trip, let's not go crazy with the drinks, okay?

G: (a bit muted) Now Trip don't get too worked up with the drinks tonight...

“GraceCounterSuggest” for TensionLow and PlayerAffinityNeutral/Trip

<set uninterruptible>

<if AgreeTrip>

T: Beautiful!

T: Perfect!

T: Great! Martini it is.

T: Alright...! Cosmopolitans!

T: Ah margaritas! Yum yum!

T: Excellent. Mmm, mmm, sangria!

T: You got it. Scotch coming right up!

<if DisagreeTrip>

T: (dismayed) No...?

T: (dismayed) What? Oh I thought you'd love that...!

T: (dismayed) Oh but we should enjoy ourselves tonight...!

T: (dismayed) Oh, but everybody loves that!

<if SpecificFancyRequest>

T: Oh, oh, yeah, that sounds even better! Great!

T: Oh yeah, great idea, great idea!

T: Oh yeah, that's even better!

T: Ooh, great, why didn't I think of that?

<if SpecificBoringRequest>

T: (let down) Oh, come on, let's enjoy ourselves tonight...!

T: (let down) Oh, but let's celebrate tonight...!

T: (let down) Oh, but let's live it up tonight...!

T: (let down) Oh, that's no fun...!

<if NonAnswer>

T: Uh, well, you know what, I'm just going to make you a martini.

T: Uh, well, you know what, I'm just going to make you a cosmopolitan.

T: Uh, well, I think I'll just make us all margaritas.
T: Uh, well, I'm just going to make us all some nice sangria.
T: Uh, well, I'm just going to pour you a glass of this wine.
T: Uh, well, I'm just going to make you a Scotch.

<if current drink suggestion is fancy>

G: No no, [Player], maybe you'd like some juice, or a mineral water?
G: No no, [Player], maybe you'd like just a simple glass of white wine?
G: No no, [Player], how about something simple, like a nice glass of chardonnay?

<if current drink suggestion is not fancy, e.g., a beer>

G: Trip that's not what [he|she] wants.
G: Trip don't force your fancy drinks on [him|her].
G: Trip don't pressure our friend, okay?
G: Trip we don't all share your infatuation with mixed drinks.
G: Trip maybe our friend isn't as excited by your suggestion as you are.

G: [Player], you'd prefer a simple glass of water, right?
G: [Player], you'd prefer a simple glass of juice, right?
G: [Player], you'd prefer a simple glass of soda, right?
G: [Player], you'd prefer just a beer, right?
G: [Player], you'd prefer a simple glass of white wine, right?
G: [Player], you'd prefer what you asked for, right?

<set interruptible>

<Trip final comment>

T: (dismayed, under breath) What...?
T: (dismayed, under breath) Oh come on...
T: (dismayed, under breath) uhh...
T: (dismayed, under breath) but... uhh...

“TripSuggest” for TensionLow and PlayerAffinityGrace

<set uninterruptible>

<if reestablish>

T: (overly earnest, a bit desperate) So! Drinks!

<if nothing suggested so far>

T: (overly earnest, a bit desperate) What can I get you?
T: (overly earnest, a bit desperate) I want to fix you something special.
T: (overly earnest, a bit desperate) Anything you want!

T: (overly earnest, a bit desperate) I'll make us something really fun,
T: (overly earnest, a bit desperate) Let's have something really fun,
T: (overly earnest, a bit desperate) We need to have something really fun,

<re-use suggestions from PlayerAffinityNeutralTrip>

<set interruptible>

<if a fancy drink was just requested>

T: (relieved, excited) Martinis... Perfect! Classic! Great idea!
T: (relieved, excited) Ah, martinis... Sweet! [Player], you've always been a classy drinker.
T: (relieved, excited) Fabulous suggestion! I'll have the same.
T: (relieved, excited) Your suggestion is perfect! I'll have one too!

<if a boring drink was just requested>

T: (nervous, wanting more) But... I'm hoping we can have some fun tonight...

T: (nervous, wanting more) But... I was hoping we'd enjoy ourselves tonight...
T: (nervous, wanting more) But... I was hoping we would celebrate tonight...!
T: (nervous, wanting more) But... I was hoping we'd live it up a little tonight...

<brag>

<use dialogue from with NeutralTrip version>

<Grace reaction to brag>

G: (confident) Trip, Trip, Trip... don't go overboard with the drink preparation tonight.
G: (confident) (slightly annoyed) Trip, you don't need to push the 'high class' drinks thing on our guests...
G: (confident) Trip, take it easy, don't make a big production out of this.
G: (confident) Trip, don't go crazy with the drinks.
G: (confident) Trip don't get yourself worked up with the drinks.

T: (overly earnest, a bit desperate) Uh, [Player], come on, what do you say?
T: (overly earnest, a bit desperate) Uh, [Player], it'll be great, what do you say?
T: (overly earnest, a bit desperate) Uh, [Player], you'll love it, what do you say?
T: (overly earnest, a bit desperate) Uh, [Player], let me make this for you, what do you say?

“GraceCounterSuggest” for TensionLow and PlayerAffinityGrace

<set uninterruptible>

<if AgreeTrip>

T: (burst of relief, a bit desperate) Ah I knew you'd agree!
T: (burst of relief, a bit desperate) Ah! I knew it.
T: (burst of relief, a bit desperate) Ahh, yes.
T: (burst of relief, a bit desperate) Ahh, I was right.

<if DisagreeTrip>

T: (dismayed, defeated) No...?
T: (dismayed, defeated) Oh, I -- I thought you'd love that...
T: (dismayed, defeated) Oh, but -- but we should enjoy ourselves tonight...
T: (dismayed, defeated) Oh, but everybody always likes my drinks...

<if SpecificFancyRequest>

T: (at first dismayed, then burst of relief) What...? Oh, oh, yeah, that's even better! Great!
T: (at first dismayed, then burst of relief) What...? Oh yeah, great idea, great idea!
T: (at first dismayed, then burst of relief) Wha...? Oh yeah, that's even better!
T: (at first dismayed, then burst of relief) What...? Yeah! Ooh, why didn't I think of that?

<if SpecificBoringRequest>

<choose one of the Disagree dismayed/defeated>

<if current drink suggestion is fancy>

G: (sweet) [Player], Trip's getting a little carried away... maybe you just want some juice, or a mineral water?
G: (sweet) [Player], I think Trip is pressuring you too much... how about just a simple glass of white wine?
G: (sweet) [Player], I think you'd prefer something simple and light, like a nice glass of chardonnay, yes?

<if current drink suggestion is not fancy, e.g., a beer>

G: (cordial admonish) Trip, darling, that's not what [he|she] wants.
G: (cordial admonish) Trip, dear, don't force your fancy drinks on [him|her].
G: (cordial admonish) Trip, Trip, you're pressuring our friend.
G: (cordial admonish) Trip, Trip, try to realize we don't all share your infatuation with mixed drinks.
G: (cordial admonish) Trip, Trip, our friend isn't as excited by your suggestion as you are.

G: (sweet) [Player], you'd prefer water, right?
G: (sweet) [Player], you'd prefer juice, right?
G: (sweet) [Player], you'd prefer soda, right?
G: (sweet) [Player], you'd prefer a beer, right?

G: (sweet) [Player], you'd prefer a glass of white wine, right?
G: (sweet) [Player], you simply want what you asked for, right?

<set interruptible>

<Trip final comment>

T: (impatient, a bit desperate) Grace... come on...!
T: (impatient, a bit desperate) Grace...!
T: (impatient, a bit desperate) uhh... Grace...!
T: (impatient, a bit desperate) but... uhh... Grace, come on...

“TripSuggest” for TensionMedium and PlayerAffinityTrip

<set uninterruptible>

<if reestablish>

T: (tense, crafty) So... drinks...

<brag 1>

T: (tense, bragging) This is great... For us I'm going to open an amazing, I mean, exquisite Bordeaux.
T: (tense, bragging) Ah I know... I'm going to open us a magnificent, I mean, astounding Bordeaux.

<local deflect dialogue: positive, negative, neutral>

T: (interrupted, smiling) Y -- yeah, I'm going to open an exquisite Bordeaux!
T: (interrupted, puzzled, brow knit) N -- no, I'm going to open an exquisite Bordeaux!
T: (interrupted, a bit puzzled) W -- well, I'm going to open an exquisite Bordeaux!

<if reestablish>

T: (tense, bragging) So... I'm going to open an exquisite Bordeaux...

<set interruptible>

<brag 2>

T: (tense, bragging) Top of the line, very rare, very difficult to acquire --
T: (tense, bragging) Best of the best, you can't buy this in stores... Very, very special --

G: God Trip, you are such a wine snob. Just like my dad.
G: God Trip, you're just like my dad with the whole wine snob thing.

T: I'll take that as a compliment. [Player], what do you say?

“GraceCounterSuggest” for TensionMedium and PlayerAffinityTrip

<set uninterruptible>

<if AgreeTrip>

T: Excellent! You've got good taste.
T: Perfect! Ooh, you're going to love this.

<if DisagreeTrip>

T: (dismayed) Oh, but this is a very special bottle of wine...!
T: (dismayed) What? Oh I thought you'd love that...!
T: (dismayed) Oh but we should enjoy ourselves tonight...!

<if SpecificFancyRequest, SpecificBoringRequest, or NonAnswer>

T: No no I really think we should have this wine, trust me, trust me!
T: Oh, come on, let's enjoy ourselves tonight, this wine will be so good. So good.

<if current drink suggestion is fancy>

G: (angry) No no, [Player], maybe you'd like some juice, or a mineral water?

G: (angry) No no, [Player], maybe you'd like just a simple glass of white wine?

G: (angry) No no, [Player], how about something simple, like a nice glass of chardonnay?

<if current drink suggestion is not fancy, e.g., a beer>

G: (angry) Trip that's not what [he|she] wants.

G: (angry) Trip don't force your fancy wine on [him|her].

G: (angry) Trip don't pressure our friend, okay?

G: (angry) Trip we don't all share your infatuation with overpriced wine.

G: (angry) Trip our friend isn't as excited by your suggestion as you are.

G: (angry) [Player], you just want water, right?

G: (angry) [Player], you just want juice, right?

G: (angry) [Player], you just want soda, right?

G: (angry) [Player], you just want a beer, right?

G: (angry) [Player], you just want a glass of white wine, right?

G: (angry) [Player], you just want what you asked for, right?

<set interruptible>

<Trip final comment>

T: (dismayed, under breath) What...?

T: (dismayed, under breath) Oh come on...

“TripSuggest” for TensionMedium and PlayerAffinityGrace

<set un interruptible>

<if reestablish>

T: (tense, desperate) So... drinks...

<grace comment 1>

G: (biting) Trip thinks he's at his classiest when he's on the serving end of a swizzle stick.

G: (biting) Trip's favorite pastime is to get the blood alcohol content of his guests higher than his golf score.

G: (biting) Trip would try to serve you one of his 'high class' drinks before even saying hello if he could.

G: (biting) My dad bought Trip a silver plated cocktail shaker for Christmas a few years back -- the rest is history.

T: (loud, angry, interrupting) Why don't I make us one of my new drink inventions,

T: (desperate, biting) I call it Grace's Inner Soul. It's a mixture of chardonnay, bitters and lots of ice.

<set interruptible>

<local deflect dialogue: positive, negative, neutral>

T: (interrupted, smiling) W -- wait, I want to make you one of my drink inventions!

T: (interrupted, puzzled, brow knit) W -- wait, I want to make you one of my drink inventions!

T: (interrupted, a bit puzzled) W -- wait, I want to make you one of my drink inventions!

<if reestablish>

T: (desperate, biting) >So... I should make you one of my drink inventions...

<grace comment 2>

G: (confident, pretending to be under breath, loud whisper) >It's a secret -- Trip doesn't even like the taste of alcohol.

T: (anxious, ignoring Grace) What?! So, [Player], how does that sound?

“GraceCounterSuggest” for TensionMedium and PlayerAffinityGrace

<set uninterruptible>

<if AgreeTrip>

T: Ah, [Player], you have deliciously wicked taste.

T: Ah, [Player], you are an adventurous drinker, like me.

<if DisagreeTrip>

T: (dismayed, anxious) No?

<if SpecificFancyRequest, SpecificBoringRequest, or NonAnswer>

T: (dismayed, anxious) You -- you don't want my invention?

<if current drink suggestion is fancy>

G: (sweet) [Player], Trip's getting a little carried away... maybe you just want some juice, or a mineral water?

G: (sweet) [Player], I think Trip is pressuring you too much... how about just a simple glass of white wine?

G: (sweet) [Player], I think you'd prefer something simple and light, like a nice glass of chardonnay, yes?

<if current drink suggestion is not fancy, e.g., a beer>

G: (admonish) Trip, darling, that's not what [he|she] wants.

G: (admonish) Trip, dear, don't force your fancy drinks on [him|her].

G: (admonish) Trip, Trip, you're pressuring our friend.

G: (admonish) Trip, Trip, try to realize we don't all share your infatuation with mixed drinks.

G: (admonish) Trip, Trip, our friend isn't as excited by your suggestion as you are.

G: (sweet) [Player], you'd like water, right?

G: (sweet) [Player], you'd like juice, right?

G: (sweet) [Player], you'd like soda, right?

G: (sweet) [Player], you'd like a beer, right?

G: (sweet) [Player], you'd like a glass of white wine, right?

G: (sweet) [Player], you simply want what you asked for, right?

<set interruptible>

<Trip final comment>

T: (very tense exhale)