SKALD: AN AFFORDANCE-BASED USER INTERFACE FOR INTERACTIVE FICTION

A THESIS SUBMITTED TO THE GRADUATE DIVISION OF THE UNIVERSITY OF HAWAIʻI AT MĀNOA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

IN

COMPUTER SCIENCE

MAY 2015

By

Zach Tomaszewski

Thesis Committee:

Scott Robertson, Chairperson
Kim Binsted
John Zuern

Keywords: interactive fiction, user interface, affordances

ABSTRACT

Interactive fiction (IF) is a long-lived text-based computer game genre. A small community of game authors are still producing independent IF games, and IF can be used for a number of serious applications, including education and AI research. This work explores the defining features of IF as a form, as well as its shortcomings in terms of user interface (UI) affordances. It then provides Skald, an alternative menu-driven UI for IF. An empirical evaluation shows that Skald offers a number of advantages over the traditional IF interface. It eliminates user input errors, is rated as easier to use, and encourages players to explore the virtual game world to a greater degree. However, the study also reveals that Skald does not significantly increase users' feelings of world-level user agency and that a sizable minority of players still prefer the traditional UI overall.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

INTRODUCTION

In the history of computer games, the text-based adventure game genre was significant but short-lived. It began in the late 1970s with *Adventure*, boomed during the 1980s, and largely disappeared in the 1990s. However, the genre remains intriguing. A small community of interactive fiction authors continue to produce independent text-based games to this day. With the rise of mobile devices, web-based interfaces, and a widening interest in digital games, interactive fiction is now within the reach of a broader audience. Interactive fiction platforms also allow for rapid prototyping of a well-defined virtual world that is useful in interactive narrative generation and related artificial intelligence research.

As a text-based medium, users of interactive fiction read descriptions of the world and the events in it and then reply by typing what they want to do next at a command line. Proponents of the genre claim that this command-line parsing of natural language input is a defining feature of interactive fiction. However, this interface is also a barrier to many modern users. In reading a description of the world, it is not always clear which of the described objects can be interacted with. It is also not always clear which verbs or commands the game supports.

Skald is a new web-based user interface for interactive fiction that uses contextual links and menus to clearly afford all of the currently-supported actions within an interactive fiction world. This approach prevents all possible syntactic and illogical user errors. By eliminating the need for a keyboard, Skald is also simpler to use on a mobile device.

This thesis presents a short background of interactive fiction and the Skald design. It then evaluates Skald's impact on users' experience of interactive fiction.

INTERACTIVE FICTION

History

The first text adventure game was *Colossal Cave Adventure*, commonly called *Adventure* or simply *Advent*. *Adventure* was written in 1975/6 by Will Crowther as a personal project to create a game that his two young daughters could play. An avid caver, Crowther modeled the "rooms" and locations of *Adventure* on the Bedquilt entrance of the Flint-Mammoth cave system in Kentucky. Sprinkled through this landscape are a few fantasy elements, such as a dwarf and a pirate, and various puzzles to solve. With Crowther's permission, the game was soon expanded upon by Don Woods, a graduate student at Stanford University. This version inspired many other *Adventure* clones, variants, and imitators across the networked mainframes of late 1970s academia (Montfort 2005).

One of the most popular *Adventure* clones was *Zork*, developed by students at MIT. Its developers soon started a software company, Infocom, in 1980. Infocom developed a virtual machine called the Z-Machine. This virtual machine allowed Infocom's game authors to code a single version of a game that could then be played on most of the diverse and usually incompatible home PC hardware available at the time. Infocom started by converting their mainframe version of Zork into a trilogy of games for PCs. However, they soon expanded beyond the fantasy-tinted dungeon-crawling genre. Their catalog eventually included murder mystery, sci-fi, and pulp fictions games (Montfort 2005).

Other companies, such as Scott Adam's Adventure International, Synapse, Level 9, Magnetic Scrolls, and Legend Entertainment soon joined the market. As they collectively explored broader genres, wrote games based on existing novels, and collaborated with established fiction authors on new games, many of these companies embraced the term *interactive fiction* (IF) as a more literary and inclusive term than *text adventure*. Text-based games were no longer only dungeon-crawling adventures (Montfort 2005).

The commercial boom of interactive fiction was short-lived, however. By the end of the 1980s, it was losing ground to the new graphical adventure games, such a Sierra's *King's Quest* and *Space Quest* series and LucasArts's *Secret of Monkey Island*. The last mainstream commercial IF game was Legend Entertainment's *Gateway II: Homeworld* in 1993 (Montfort 2005).

Yet the end of IF's commercial viability did not mean the death of the form as a whole. To this day, a small but avid community of independent developers continue to produce

interactive fiction games. Originally formed on the Usenet newsgroups rec.arts.int-fiction and rec.games.int-fiction, this community has hosted an annual Interactive Fiction Competition (2015) since 1995. The community includes such developers and authors as Graham Nelson, Andrew "Zarf" Plotkin, Roger Firth, Adam Cadre, Emily Short, Mike Roberts, and Eric Eve. The community maintains a public archive of decades of IF games in the IF Archive (2015) and the Interactive Fiction Database (2015). Sites such as the *Brass Lantern* (2015) help introduce both players and new authors to the genre.

Many of the artistically significant IF games have appeared from this community, well after the commercial boom. Examples include such games as *Galatea*, *Photopia*, and *Lost Pig*, though many other examples also exist. Emily Short's *Galatea* (2000) subverted the normal puzzle basis of IF by providing only a single room and a living statue to converse with. Adam Cadre's *Photopia* (1998) also does away with most puzzles. It consists of a series of seemingly disconnected short vignettes, some fantastic and some mundane. Cadre uses color to mark the different fantasy sections. The stories all come together into a single poignant conclusion that reveals that the different stories all revolve around a single teenage girl whom we never play directly as a character. *Lost Pig* (2007) pokes fun at the traditional puzzle-solving dungeon-crawling genre of IF while exploring the effects of narrator voice.

For a much more extensive history and introduction to interactive fiction, see Nick Montfort's *Twisty Little Passages* (2005).

Essence of IF

Traditional interactive fiction can be defined as *an explicitly-modeled narrative world with a text-based user interface*.

Thus, the first essential feature of IF is an *explicitly-modeled world*. By "explicitly-modeled", I mean that the relevant states of objects in the virtual world are tracked using variables in computer memory. For example, a game may contain a desk drawer. Not all possible features of this drawer will be modeled. For example, the color of its knob or its exact measurements may not be relevant to this particular narrative. However, some state about it will be tracked. At the very least, this state will include the object's current description and what actions can be performed on it. Additional state might include its location: the drawer is in the desk, which in turn is in the library. The model might include whether the drawer is currently open or closed, and it may include which other objects are currently located within the drawer. Furthermore, most of these states will need to be conveyed to the player through the description of the object. That description will then need to change accordingly whenever the state of the object changes—such as if the player opens the drawer or pulls it out entirely and carries it off with him (if this hypothetical drawer supports such an action).

Secondly, an IF world fills some sort of *narrative purpose*. This narrative may often be fairly primitive, consisting of little more than the entertaining series of events that is produced by the player's exploration from the first location in the game through the last. Yet there is often both a clear starting and ending point in IF games. This gives us the essential beginning-middle-end structure of a story (Aristotle 1961). In addition, there are often puzzles and challenges for the player to overcome along the way that impose a certain linearity to the middle of the story. For example, before the player can enter a castle, she may have to climb a tree and discover the key to the front gate in a bird's nest. It is this combination of challenge and narrative that makes IF a game, rather than only a virtual world simulation.

Finally, the user interface to an IF world is *text-based*. The world—including both objects and events—is revealed through text descriptions. Also, input to an IF game is text-based. Early games typically supported only one or two-word imperative commands, such as verb or a verb and the object to be effective. Most modern IF games offer a robust natural language parser that can handle sentence-like grammar and differentiate between ambiguous objects based on the current world state.

IF games are usually turn-based. A user reads the current description, types in a command, and is rewarded with a narration of the effects of that action (or an error message, if the command could not be interpreted as a valid action). The IF system then waits for another input from the user.[1] This cycle of turn-taking produces an ongoing dialog between the IF system and the player.

Given this definition, there are a number of different angles from which we can conceptualize interactive fiction. First of all, IF is a computer program, with such corresponding features as input, output, procedural generation, and saving state from memory to disk. Secondly, we can examine IF in terms of its virtual world, such as its objects and behaviors. Alternatively, we can think of IF in terms of a game, exploring its challenge level, rewards, scoring mechanism (when present), and user enjoyment. Specifically, we can explore the puzzles that comprise the bulk of the material of most IF games.

As its name suggests, narrative is also a very important aspect of any interactive fiction. This includes the unfolding plot, the various characters, the quality of the narration, the richness of the descriptions, the mood or tone, the genre, and so. The player may piece together a past story through notes or clues left behind by a previous adventurer or a lost civilization. The story may be fairly nascent, emerging primarily from the player's exploration of an open world. Or there may be a well-defined tale, given a tight linear structure by the preconditions and necessary sequence of the puzzles that make up the game.

1 Some IF games do have real-time events. In these cases, new events and output may occur even if the user does not enter another command.

Nick Montfort (2005) offers the metaphor of a literary riddle as another way to conceptualize the nature of interactive fiction. Like IF, riddles have the potential for significant literary quality or to give us cause to reflect upon our understanding of the world. Yet, like IF, riddles are often marginalized as only trivial games for children. The riddle present an implicit puzzle to be solved. It casts the world and the objects it describes in certain terms that the solver of the riddle must explore closely in order to proceed. For example, here is a classic riddle:

> *What runs but never walks,*
> *Often murmurs but never talks,*
> *Has a bed but never sleeps,*
> *Has a mouth but never eats.*

It casts a common object[2] in terms of human qualities. It implicitly asks to listener to explore that description and propose different solutions until one—often obvious in retrospect—is finally discovered. This is not so different from playing an IF game.

Related Forms

When defining a new category, it is often useful to explore the boundaries and edge cases of that category. There are a number of other game forms that are similar to IF in some way but that are not considered to be interactive fiction in the traditional sense.

Hypertext fiction is also a digital text-based genre. Indeed, both IF and hypertext fiction are often grouped together under the broader category of digital literature. Works of hypertext fiction comprise of a number of separate pages connected by links. Such works can be implemented to run in a special hypertext interpreter—such as Storyspace (Bolter & Joyce 1987), which predates the World Wide Web—or simply as HTML pages in a web browser. The reader clicks on links embedded within the text to move between the pages. In some hypertext works, the path taken by the reader through the pages—also called *lexia*—does not affect the events of the narrative. Instead, it affects only the discourse-level experience by changing the order in which the lexia and related story events are experienced. In other hypertext works, the links chosen by the reader may actually affect the underlying story by taking the reader through only a subset of the possible lexia, much like a *Choose Your Own Adventure* book.

Unlike interactive fiction, hypertext fiction does not explicitly model the story world as software objects in computer memory. Therefore, it cannot track changes in the state of that world. Also unlike traditional interactive fiction, there is no natural language parser for input.

---

2 A river.

Some hypertext fiction systems, such as Twine (Klimas 2015), muddy these waters a bit by providing support for variables and conditional behavior. Although this still does not provide a full world model, it does allow for some limited tracking of world or story state. This is somewhat reminiscent of the *gamebooks* that were popular in the 1980s. These paperback adventures added initial tool and skill selections, dice rolls, and randomized combat to what was otherwise a *Choose Your Own Adventure* story. Joe Denver's *Lone Wolf* is a classic example of this genre and is now available in digital form (Project Aon 2014). While I would argue that Twine is not true IF, the distinction gets fuzzy and others disagree. Many Twine games are now listed in the IFDB (2015).

Graphical adventure games evolved directly from text adventure games. Both have an explicitly modeled world. *Mystery House* (Williams 1980) was essentially a text adventure game, but with shorter room descriptions and a single line-drawing for each location. Yet this game lead to the creation of Sierra Online. As their graphics improved, graphical adventures could dispense with the room description entirely. While these maintained a natural language parser through most of the 1980s, they eventually transitioned to a point-and-click system for input. Legend Entertainment's games also pushed the boundary of IF toward graphical games, though they essentially remained IF games.

MUDs (Multi-User Dungeon) and MOOs (MUD, Object Oriented) are very similar to IF and were in fact directly inspired by *Adventure*, *Zork*, and other early text-adventure games. As the name suggests, many early MUDs persisted the fantasy-based dungeon-crawling adventure theme, though later MUDs did explore different genres and even supported non-game-related environments for social chat and education. Like IF, MUDs are primarily text-based for both input and output and they have a modeled virtual world. The difference is that MUDs are multiplayer, with players connecting to the MUD server over the Internet or (in the past) by modem. The social aspect of MUDs tends to shift the focus of the game experience away from narrative, lone exploration, and puzzle-solving towards chat, roleplaying, and player vs player combat or collaboration. As with the evolution from IF to graphical adventure games, MUDs gradually included more graphical elements and eventually evolved into MMORPGs (Massively Multiplayer Online Roleplaying Games).

Finally, there are text-based world simulation systems like SHRDLU (Winograd 1971). SHRDLU was an early AI program that explored natural language understanding. Like an IF game, it included a detailed virtual world and a complex parser for input. Its output included both text and an image of the current world state. Unlike IF, the primary goal of SHRDLU was not entertainment or to produce a narrative or game-like experience.

Current IF Platforms

Although a number of others exist, the two most popular platforms for authoring interactive fiction today are Inform ("Home: Inform" 2015) and TADS ("TADS" 2015).

Inform

Inform has the larger community. Inform games can compile into one of two formats. One runs on the Z-Machine virtual machine, which was designed by Infocom in the late 1970s. The other supported format is the Glulx virtual machine, designed by Andrew "Zarf" Plotkin in the late 1990s ("Playing Interactive Fiction" 2015).

Since Inform games are designed to run in a virtual machine, an interpreter is required to run them. While most of these are designed for major operating systems such as Windows, Mac OS X, or Linux, a number of other platforms are also supported. There has recently been work done to develop browser-based interpreters. For example, Parchment ("Parchment" 2015) supports both Z-Machine and Glulx games. Quixe (Plotkin 2014) is a Javascript-based interpreter for Glulx files.

Inform 7, the latest version of Inform, has significantly changed the Inform programming language to be more like natural language. It also offers an Integrated Development Environment (IDE) and other programming tools.

Aside from the compiler and interpreter, the Inform community has also created a number of code libraries and extensions that a game author can leverage in their games for added functionality.

TADS

Like Inform, TADS—the Text Adventure Development System—has been in active use since the early 1990s. TADS games also compile to a custom virtual machine architecture. The TADS language has a C-like syntax, and it includes a robust default library for modeling complex objects and the interactions between them. TADS3, the latest version, includes a client-server architecture for running web-based IF games.

Serious Applications of IF

Interactive fiction has been used for more than entertainment. It is has been used widely in education, from middle school through college level courses ("Teach with Inform" 2015). At lower academic levels, it can be used to teach reading and foreign language skills. For young authors, it can be used to teach programming basics. In more

interdisciplinary programs, students can author games as projects to learn history, narrative, game design, and new media concepts.

IF in Interactive Drama Research

Within the broader field of interactive narrative, interactive drama research explores how artificial intelligence might be used to author more interactive stories. Specifically, in an interactive drama the player assumes the role of a character in the story. An AI agent then acts as a drama manager for that story, actively shaping or even creating the story at run-time in response to the player's actions in the virtual world.

Interactive fiction provides an appropriate medium for such works, especially in prototype stages. Out of the box, Inform and TADS provide the tools to quickly and easily build a sufficiently complex story world, including simple characters. The existing interpreter handles text input, and text as an output medium can support a rich, emotional experience without the time and expense required for detailed graphics or physics engines. The following are some examples of such uses of IF.

In evaluating a search-tree-based approach to interactive drama management, Weyrauch (1997) constructed a short text-based scenario named *Tea for Three*. This scene was inspired by part of Infocom's 1982 game, *Deadline*. Evaluation was done using simulated players in order to generate a high volume of sample game runs.

Nelson and Mateas (2005) revisited Weyhrauch's particular search-based approach, applying it to a modified subset of Michael S. Gentry's IF game, *Anchorhead*. Finding that Weyhrauch's approach did not perform as well on a different game, Nelson et al. proposed a declarative optimization-based drama management (DODM) approach (2006). In a DODM system, the author defines the story as a series of high-level plot points and provides an evaluation function to rate different possible plot-point sequences. The drama manager then uses this evaluation function and a predictive model of future player actions in order to make changes in the game world that are likely to make particular future plot points more or less likely. In this way, the drama manager can guide the player towards a story that more closely fits the author's original vision. Evaluation of this approach was done using simulated players.

Sharma, Ontañón, Mehta, and Ram (2010) used a different approach to drama management. Their case-based drama manager uses models of past players' story preferences to determine optimal story paths. By comparing the current player's actions to those of its models, it attempts to encourage the player towards subplots deemed to be of greater likely interest while steering the player away from plots of lesser interest. Sharma et al. used the same portion of *Anchorhead* used by Nelson et al., but they

tested their system with live players. They argued that such evaluation offered much richer findings than using simulated players.

In 2011, I used Inform to author an IF game, *Demeter: Blood in the Sky*, in order to evaluate my own approach to interactive drama management (Tomaszewski 2011). My drama manager, Marlinspike, used pre-authored story segments called scenes to advance the story in response to player actions. Marlinspike would either select the next scene or customize its contents in such a way as to build upon, or reincorporate, as many of the previous player's actions as possible.

Like Sharma et al., I tested my system using using real players. I also found this to be a valuable approach. One of my discoveries was that, while system-level metrics showed a significant improvement to the resulting story structure when Marlinspike used reincorporation versus when it did not, real players did not report a corresponding difference. However, I also discovered that many players had difficulty with the traditional IF interface. Distracted by input errors and unclear affordances, they seemed to lack the world-level agency required to engage effectively with the story-level changes Marlinspike was producing.

Given the history of IF's use in interactive drama and the growing realization that new systems need to be evaluated with real users, I became interested in improving IF's traditional user interface.

AFFORDANCES

While it is valuable to explore interactive fiction in terms of its content—to conceptualize it in terms of *game* or *narrative* or *puzzle* or *riddle*—works of IF are computer programs first and foremost. While an IF author hopes that the program's user interface is transparent enough to "fall away" and let the user engage directly with work's content, this is not always the case.

Limitations of IF's Command Line Interface

IF's traditional user interface provides a simple command line prompt at which the user can type what they want to do in natural language. Often, these are simple imperative commands, such as `look` or `wait` or `get lantern`. Or they can be much more complicated, such as `tell sally to take the broken glass and put it into the red bin`.

However, while the potential variation in natural language utterances is incredibly broad, works of interactive fiction are necessarily finite in scope. When the interface suggests that any logical command will do—yet the system itself is authored to handle only a subset of what the user might think to try—there will eventually be a jarring disconnect. This disconnect between a user-entered command and system-supported interpretation can occur for a number of different reasons.

First, the IF parser might support the particular action that the user is attempting but simply lack the particular word form or synonym that the user is giving. For example, the user might enter the command `chat with the stable boy` and get an error message: `I don't know the word "chat"`. The user's goal here is to have a conversation with the stable boy. The game may well support the action of conversing with other characters, and it might map a number of different synonyms—such as `talk`, `gossip`, and `converse`—to this same basic *talk* verb. But because the game lacks the "chat" synonym, the user's command fails.

This same synonym problem affects nouns as well. For example, if an IF author neglects to indicate that a stone bust can also be called a "statue", the command `get statue` will fail while `get stone bust` will work.

Similarly, the parser may support only certain grammatical structures. `Give egg to troll` might work, while `give troll egg` or even `give troll the egg` may fail.

In addition to such syntactic mismatches, an IF parser can fail at a semantic level. The easiest way this can happen is if the IF world does not support the given verb under any synonym. For example, a player might want to `repel the vampire`, but this *repel* action is simply not possible in the game if the IF author has not written the code necessary to support it.

Similarly, objects that should logically be present or that are even mentioned briefly may not actually be modeled in the virtual world. For example, the player's character might be standing on a busy urban sidewalk with an urgent package to deliver under one arm. Yet if the player pauses to type `look at street`, she might get the response `You see no such thing`. Logically, if there is a sidewalk, there should be a street, but the author may not have modeled it explicitly. This disconnect can be compounded if the author actually mentions traffic, cars, or even the street itself in the description of the location.

This failure can be slightly less jarring if the error message is crafted in such a way that the IF parser appears to recognize what the player meant but is simply reporting that the player is exploring in an unsupported direction. In the example above, the parser could instead reply with `The street is unimportant` or `You do not need to refer to the street in this game`. The edges of the game are still revealed to the player here, but not as sharply. It implies that the street was not forgotten by the author, but deliberately not implemented.

As with objects, there is a range of ways to recognize but still not support particular user actions. One is to provide only illusory success: some outcome that has no effect on the game world. For example, if the user types `jump`, the reply might be `You jump up and down on the spot`. The system has recognized and accepted the user's command, but, beyond the few lines of code needed to provide this short response, the IF author does not need to implement any world state changes that result from jumping.

Alternatively, an IF system can refuse a player's request. Suppose the player types `get statue`. If the statue is simply decoration and not essential to the author's story, the author will not want to spend time modeling all of the possible interactions between this statue and other objects and locations in the game. There is no obvious way to provide illusory success here: either you can carry the statue off with you or you can't. This means the author has to refuse the player's command. This can be a blunt refusal: `You cannot get the bust`. It could accept that the attempt makes sense, but explain why it fails, either for a physical reason (`The stone bust proves too heavy to lift, let alone carry around with you`) or by reinterpreting the player character's motivation (`Your troubles are heavy enough. You decide not to add the extra burden of the stone bust`). Finally, some refusals may be

hints that the action is possible but by a different means. For example, the refusal `You can't reach the stone bust from here` implies that it might be possible to reach the statue later given the use of a stool or ladder or rope.

An IF author has a particular story and world in mind. Given her limited time and resources, she will focus on authoring objects and actions necessary to that story. However, when this finite game scope is paired with the apparently unrestrained user interface of a natural language command line interface, the user will eventually attempt an interaction that is beyond that scope. If this disconnect occurs at a syntactic level, the parser will be unable to make sense of the command at all, even if it would have understood synonymous commands. Even if the parser can make sense of the command, it may still need to be refused more or less elegantly at the semantic level, unless the author is willing to spend the time and energy to fully support and debug all the possible outcomes of supporting the action. For this reason, much of an IF author's time goes into crafting humorous or gentle refusals, attempting to provide hints that guide the user back to the actions and objects that are actually important to the game.

This disconnect ultimately stems from the opaqueness of the interface. A simple text prompt suggests that it can accept a free range of input. However, it does not indicate the supported command grammar. It does not indicate the valid synonyms for either verbs or game objects. An IF game's description can often fail to indicate which objects support a wide range of actions versus those that don't.

A different UI could resolve many of these limitation by clearly indicating interactive objects and the exact range of supported actions for each, all without the possibility of syntactic input errors.

Interface and Narrative Affordances

Clear affordances have been the foundation of human-computer interaction (HCI) and user interface design for decades.

The concept started with psychologist James Gibson (1977). For Gibson, an affordance is an "action possibility" offered by particular features of the environment. For example, a smooth clean level surface a couple feet off the ground affords *sitting* to most humans. If the surface has a long smooth edge, it might also afford *grinding* to a teenager with a skateboard. Although an affordance is a relationship defined by the capabilities of a particular actor, Gibson still considered affordances to be based upon objectively measurable features. The environment could still afford a given use even if this potential went unperceived and unused by a given actor.

In 1988, Donald Norman (2002) applied this concept to interaction with human-constructed objects. From here, the concept was taken up by HCI. Norman focuses much more on perceived affordances. That is, due to an actor's capabilities, goals, plans, or past experiences, the environment can suggest some possible actions more strongly than it does others. This emphasis on perceptibility is valuable to system designers, since a possible-yet-unperceived use of an object or the environment is ultimately of little practical utility. This is the sense of *affordance* that I will use here.

Norman suggests that there are seven stages to performing a significant action in the world. The actor 1) forms a goal, 2) breaks this goal down into specific intentions and 3) translates those intentions into necessary actions in the world that are then 4) executed. The actor then 5) perceives and 6) interprets the state of the world in order to 7) evaluate the outcome relative to their original goal.

A system's affordances play an essential role in this action process. The actor must understand the general function of a given device, recognize how this relates to her goals, and then determine which specific actions are possible with it as related to her intentions. If the actor's intentions cannot easily be mapped to actions supported by the system, there is a significant Gulf of Execution. Similarly, if it is not clear what effect a given action had on the system, there is a significant Gulf of Evaluation.

Norman's approach to affordances has been applied specifically to the domain of interactive drama. By blending the narrative poetics of Aristotle with the concept of affordances, Mateas (2004) proposed a model of user agency for interactive drama. Specifically, the various possible actions a user might make provides the "material cause" for a user's interactions. At the same time, the context of the story-so-far provides suggestions as to what goals or intentions might be most appropriate. These story constraints provide a "formal cause". Mateas proposes a user will feel the most agency when these two causes are balanced—they have one or more goals in mind thanks to the story context, and the interactions possible in the virtual world sufficiently afford the means by which those goals might be realized.

Tomaszewski & Binsted (2006) expanded this model to multiple levels of narrative. For example, it is possible to differentiate world-level from story-level agency. World-level agency is the ability to successfully interact with the virtual world. The game's medium provides a representation—such a text description or graphical depictions—of that virtual world. From this raw material, the user can mentally construct the existence of particular game world objects. The player's constructed model of that virtual world then provides the context and constraints for determining which object interactions should be possible within the world.[3] We might use the term *verb* to refer to these world-supported

3 This is essentially the process described by Norman (2002) through which a user develops a mental model of a given system based on the system's observable system image.

interactions. The game's medium also affords the input controls by which the player can then invoke the desired verb on a given object.

Such world-level agency is a prerequisite for any story-level agency. For example, a story's context might provide the player with the story-level goal of courting another character named Sam. The player's conception of the world objects and the possible verbs then provides her the material for such action. For example, she might try to compliment Sam in conversation, give him a gift, or defend him from the attacks of other characters. To do so, the player would then need to exert her world-level agency to actually perform one of these specific actions in the game world. These world-level deeds may then produce the story-level effect of Sam thinking more highly of the player's character. However, in an interactive drama, the player is not usually offered a chance to interact directly at the story level, such as by typing `make Sam like me more`.

Previous Research Regarding IF Affordances

Some of the challenges users face when dealing with traditional command-line input in a story-oriented or textual virtual world have already been empirically examined to some extent.

In Tomaszewski (2011), I evaluated an IF-based interactive drama. 24 participants completed the study. Despite each player completing a 10-minute in-world IF tutorial before playing their first recorded session, 20% of entered game commands failed to produce a valid response from the system. In open comments, 25% of participants listed the command-based input system as the least enjoyable aspect of the game, while only one participant mentioned it as the most enjoyable aspect.

Mehta, Corradini, Ontañón, and Henrichsen (2010) explored the tradeoffs between graphical and text media for game output and between menu and natural language modalities for input. To do this, they used the same portion of *Anchorhead* used in earlier DODM research (Nelson 2005, Sharma 2010). Their study did not clearly separate these two concerns, however. They tested a graphical version of the game with natural language input against a text-based version with menu-driven input. When examining the qualitative responses of 30 participants, they concluded that the natural language input was more natural, leading players to feel more immersed and engaged. However, this input modality also "made it difficult for some users to figure out appropriate actions". It also created "a false illusion" on what range of input the system could handle without generating errors, which led to some user frustration.

Sali et al. (2010) examined the effects of different input systems for character dialog within graphical games. Specifically, they looked at using a sentence-selection menu

(where a character's exact words are shown), an abstract response menu (where the general sentiment or goal of an utterance is given), or natural language input. Using the same underlying game (Mateas and Stern's *Facade*), they implemented the three different input systems. The two menu-based systems would pause the game, while natural language input had to be entered in real time. Based on the semi-structured interview responses of 35 participants, they examined a number of different factors.

Participants were fairly split between whether natural language or one of the two menu systems led to more engagement. Some preferred the low-level control of being able to choose their own words, while others appreciated the high-level control on the game offered by choosing well-crafted and effective utterances. The natural language input system was clearly deemed the most difficult to use and the one that offered the least control. However, it was also deemed the most enjoyable by half of participants, with 49% of participants preferring it over either of the two menu systems.

Command Line Augmentations in IF

In the late 1980s and early 1990s, Legend Entertainment tackled some of the affordance issues inherent to the traditional IF interface. Their games were clearly interactive fiction, but with many graphical additions. Some games included video cut-scenes or animated images of the current location. In terms of the UI, most Legend games included a location image, a compass rose, and menus of possible actions and available objects. All of these additional UI components could be interactively clicked to perform certain actions.

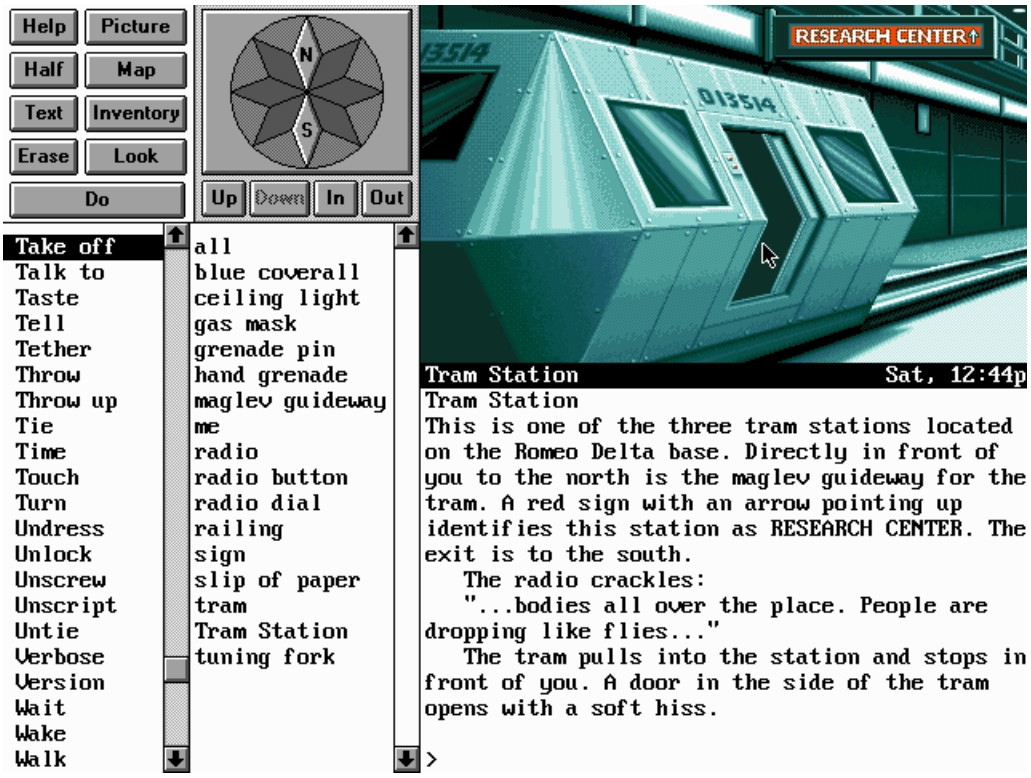

Figure 1: Screenshot from Gateway II: Homeworld

My experience of these games was that the extras were a bit like training wheels: a good way to learn what's available, but eventually I left them behind and relied on purely textual input, which was much faster. The compass rose was particularly useful, but the menus generally involved too much scrolling.

Various interactive fiction authors have continued to experiment with different IF user interfaces, often blurring the boundary between interactive fiction and hypertext fiction, as well as adding other novel textual effects. For example, some have used menus for conversations, where users select a particular utterance. Some have added hypertext links for a subset of possible actions, or have suggested a subset of possible actions as hyperlinks while still leaving the command line free for other options.

Most of these experiments have be specific to one or two particular games, however, rather than a generic solution that could be easily reused for a wide variety of IF games. Furthermore, to my knowledge, none of them have been empirically analyzed regarding their effectiveness or their effect on gameplay across different games.

SKALD

Skald[4] was designed in 2012 following the interaction design process described by Cooper, Reimann, and Cronin (Cooper 2007; Tomaszewski 2013). My goal was to design a user interface (UI) for interactive fiction that:

- prevents syntactic and illogical user-input errors,
- clearly reveals which objects in the modeled world are interactive, and
- affords the specific actions currently possible using each of those revealed objects.

A web-based or mobile-friendly interface was also an important, although secondary, design goal.

Considered Designs

I considered a number of alternative designs.

Action and Object Lists

Similar to Legend Entertainment's UI, this design would present two always-open menus: one of all the game-supported verbs and another of all the currently available objects. Clicking the words would add them to the input field.

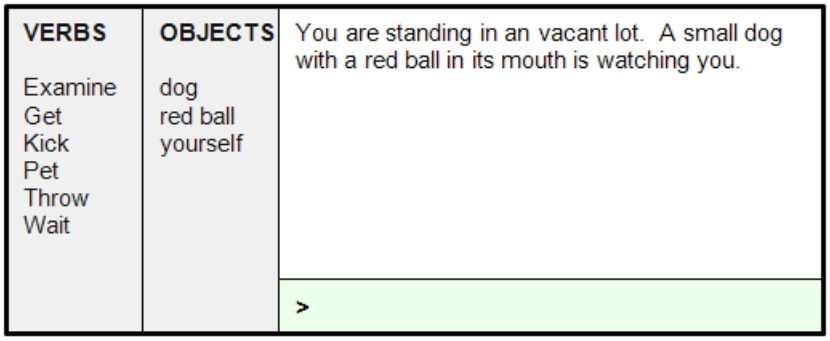

Figure 2: Prototype of an always-open menu UI

4 Skald was original envisioned to become part of a larger system named MIDGARD—the Modular Interactive Drama Game Architecture for Research and Design. *Midgard* is the human-inhabited world in Old Norse mythology. A *skald* was a Norse poet and bard. This seemed an appropriate name for the user interface responsible for conveying a MIDGARD story to the end user.

This design clearly reveals all interactive objects and all interactions. However, because not every verb-object combination is logically possible, it does not clearly afford *only* the currently-possible interactions. In addition, based on my own experience with Legend's interface, such menus or lists are tedious to use to construct commands due to their length and the scrolling required in larger games. They do still provide useful training wheels to reveal available options. When playing Legend games, I would refer to the menus for hints when stuck, but I still typed at the command prompt to input most commands.

Real-Time Parsing

It is possible to keep the command line interface yet still prevent syntactic and even illogical input errors. In its simplest form, this design would parse the user's input while they type. While the command is unparsable or represents an action that is not currently possible, the input field could be highlighted red and the input prevented. Once the command is valid, the input field could switch to green and allow the user to submit the command.

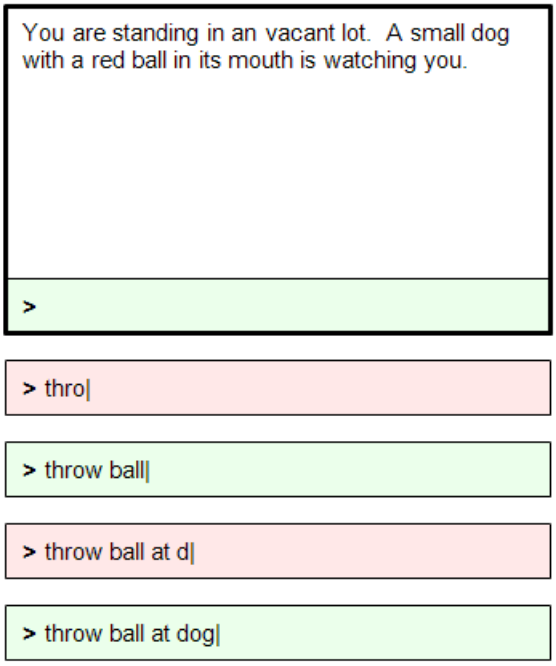

Figure 3: Prototype of a real-time parsing UI, showing different states of the input field over time

Although this design prevents invalid input, it does not afford possible commands. The user may still only discover the edges of the game through trial and error.

Inverse Parser

As an extension of real-time parsing, it would be possible to recommend a list of permissible tokens at each point of command construction. For example, the input field could start with a drop-down menu of all currently-possible verbs in the game. Then, based on the initial verb chosen, another menu could appear that contains only valid direct objects for that verb.

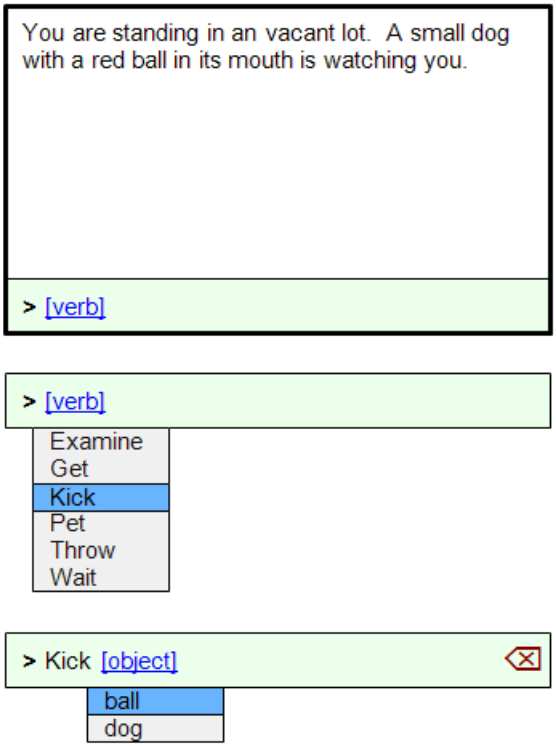

Figure 4: Prototype of an inverse parser UI, showing different states of the input field over time

This technique prevents invalid input and affords only valid commands. However, it seems that it would not be any faster than typing. Rather than a series of separate menus, it seems that it would be faster and smoother to provide a single verb menu with sub-menus for any direct objects relevant to that verb. A single final click would then produce a valid command.

Object List

Users may not always begin their thought-process knowing what action they want to take. Instead, they may wonder what objects are currently around them, or they

may be intrigued by a particular object and then wonder what they might possibly do with it. In such situations, it may be more natural have a menu of currently available objects. The user can then select an interesting object and pull up a list of the actions possible on that particular object.

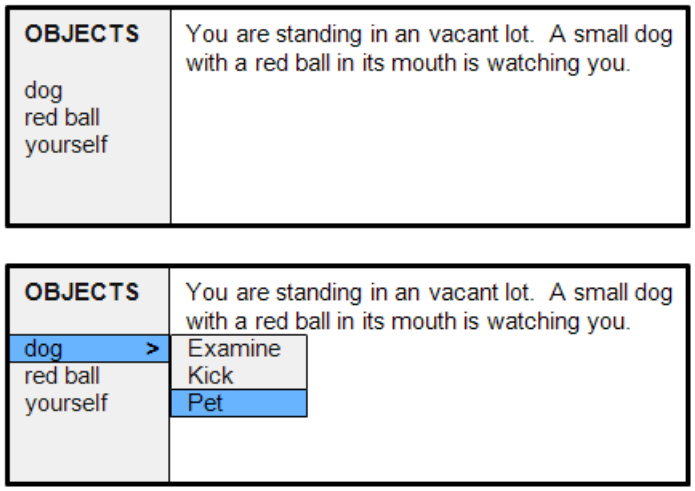

Figure 5: Prototype of an object-menu-based UI, showing different states of the menu over time

However, this design alone does not easily support actions that take no direct object, such as `Look`, `Sleep`, or `Wait`.

In-text Pop-up Menus

All of the previous designs have provided a means of input that is divorced from the primary output of IF: the text descriptions of the world. To blend these two, I could highlight or underline the names of interactive objects within the text itself, very much like hypertext links in a webpage. The user could then click on objects in context and select actions only relevant to that object.

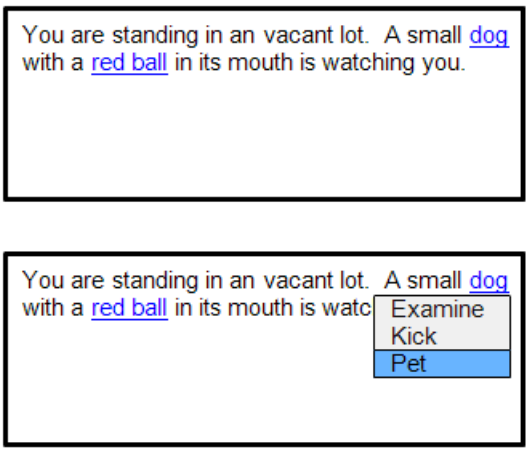

Figure 6: Prototype of an inline pop-up menu UI, showing different states of the menu over time

However, as with the Object menu design, this approach does not support verbs that take no direct object. These verbs could be afforded through a different means, such as a row of buttons along the bottom of the screen.

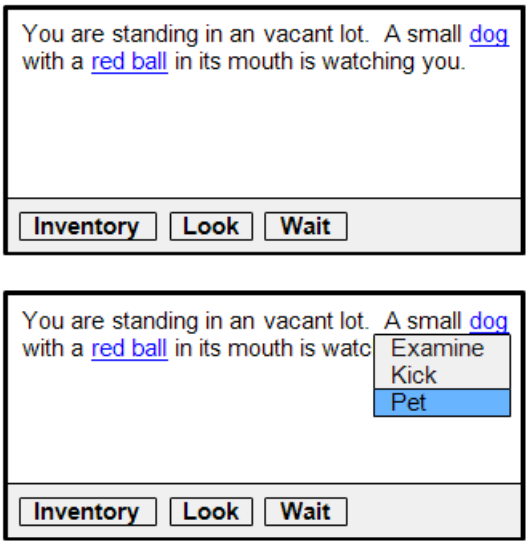

Figure 7: Revised prototype of an inline pop-up menu UI including extra verb buttons

Additionally, because the state of the game world potentially changes with each user action, it may be necessary to change previous object menus if an object changes or even disable the menu entirely if the object disappears. For example, suppose a scene describes a small cake. The user then clicks on the "small cake" and selects an `Eat cake` action. This produces a system reply that describes the outcome: "You eat the small cake." If the cake is now consumed, it should not be possible to click on the cake and examine it or try to eat it again. Indeed, any past description of the room that even mentions the cake is now only historical context. Similarly, if the player closes a door, the door's action menu needs to be updated to drop the `Close door` action and insert an `Open door` action. The simplest way to handle these various discrepancies between the state of the world as described in past turns and the current state of the world is to simply remove the text of past turns or else disable it in a way that indicates that it is no longer relevant.

Paper Prototype

Of these various designs, the in-text pop-up menus seemed most natural and unobtrusive. I constructed a paper-prototype that described a simple *Sleeping Beauty*-like scenario. The player finds himself, clothed in a torn cape and carrying a used sword, in a tower room with a sleeping princess and a songbird in a golden cage. The goal of this short scenario was to wake the princess... although it turns out that the solution is not the traditional kiss. I asked a handful of users to walk through this scenario, selecting underlined "links" on the page, and waiting for me to provide the resulting in-text pop-up menus as Post-It notes.

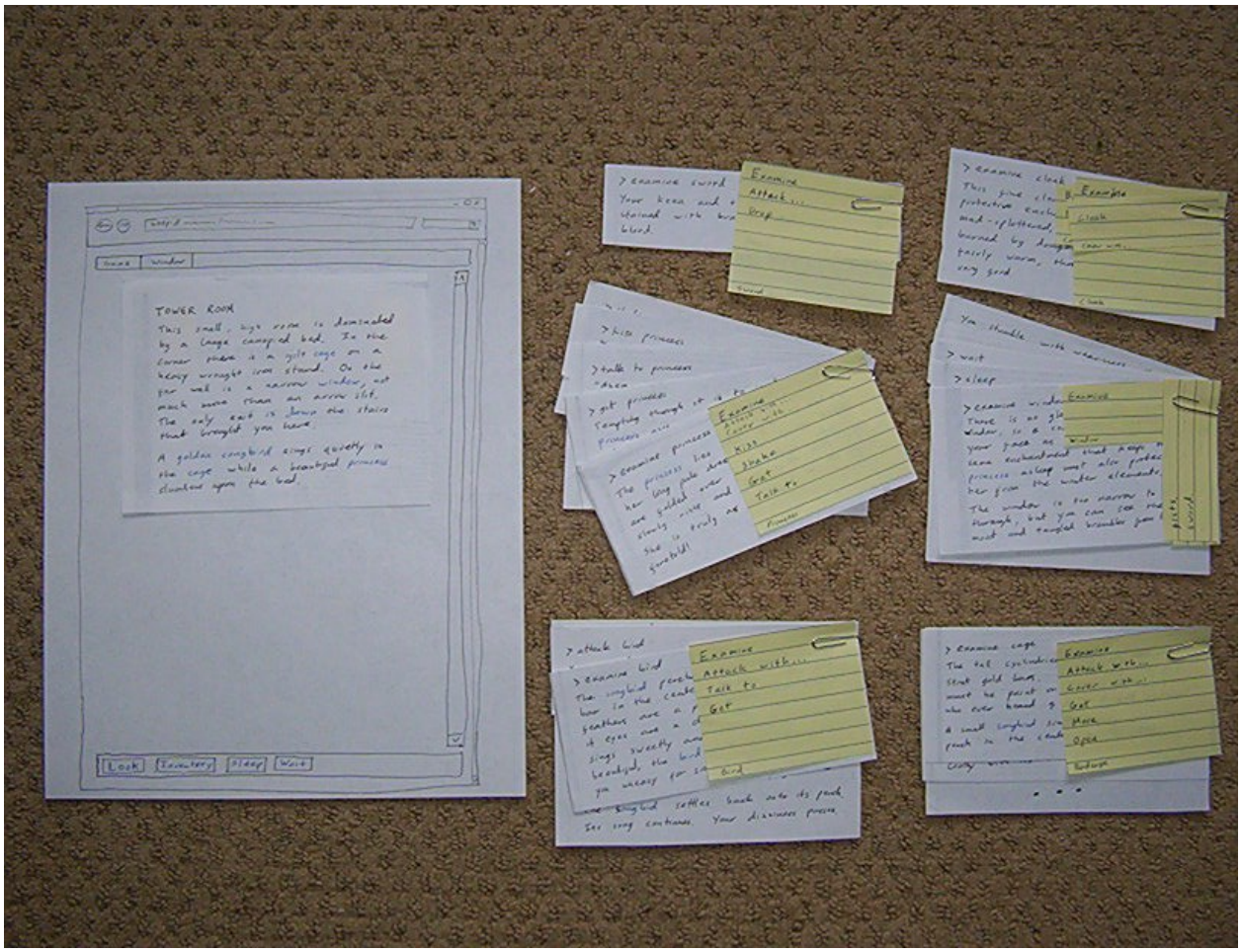

Figure 8: Skald paper prototype

This prototype revealed two interesting consequences of the in-text pop-up menu design. The first revelation was that not all of the currently-accessible objects will always be mentioned in the most-recent turn's output. Even if the description of a new room exhaustively lists all of the interactive objects it contains, it would be unusual and tedious to also repeatedly mention the player and all of the items she might be carrying in her inventory.

Furthermore, even if all room descriptions were exhaustive, most turns produce descriptions of only specific objects. For example, suppose the player enters a tower room that contains a number of different objects, including a cage. If the player then selects to `Examine the cage`, this would disable the current room description and produce a response containing only a description of the cage and its contents, without mention of most of the other objects in the room. Similarly, other actions may not

provide much description at all. For example, `close door` often results in a message as simple as `The door is now closed.`

System responses like these that contain no object links—or at least no objects that interest the player—result in "dead ends" of interaction. If previous turns' links have been disabled (for the game-state reasons described above), the user is left with no in-text means for further interaction. All is not lost: the user must simply `Look` in order to re-examine the current room and all of its described contents and exits.

There are work-arounds that would prevent this need for the user to break out of interactive dead-ends by pressing a *Look* button. For example, it would be possible to add two additional panes to the user interface: one *Location* pane that lists the complete description of the current room and its objects and another *Inventory* pane that lists the player and all of the items she is currently carrying. These additional panes would need to be automatically refreshed after every turn. The drawback here is that this design change would complicate both the user interface—which now has three separate panes of text to show the current location description, an inventory list, and the result of the most recent action—as well as the technical implementation—since each turn now requires three different responses from the system rather than only one.

While this redesign would ensure that the user always has an in-text object to interact with somewhere on the screen, it still does not provide for all possible interaction as in-text links. As mentioned previously, some game verbs such as `Look`, `Wait` or `Sleep` do not take direct objects. These actions must either be provided through a separate means—such as a row of buttons beneath the screen—or else be rather awkwardly associated with the player character object mentioned in the inventory list. This second approach would make these simple commands hard to find for new users.

Therefore, the prototype tests suggested that some supplemental means beyond only in-text pop-up menus would be needed to effectively reveal all of the currently possible actions and interactive objects at all times.

The second revelation provided by the prototype was that a new design may significantly change how users explore and play an IF game. In traditional IF, it is often necessary to closely read the text to determine what objects are present. In the prototype, object names were clearly underlined. In traditional IF, it is then necessary for the player to consider what the current puzzle or obstacle is and how she might overcome it. The player's conception of the current puzzle or goal may not be particularly clear; the player might simply explore different actions as they come to mind to see what is possible. In the prototype, these possible actions have already been delimited. Rather than visualizing and exploring the virtual world, prototype users appeared to be exploring the menus. They would click on an interesting object, scan the

resulting menu, and then select the most interesting action without much thought given to their current goal in the game. At least one user was surprised to discover that the prototype game was over, as if she had simply stumbled upon the solution.

Many traditional IF games are puzzle-based. By analogy, those puzzles are often like riddles or short-answer questions on an exam: they pose a problem to the player and then provide an open space for the player to explore a possible solution. A user interfaces that affords all possible actions effectively turns those problems into multiple choice questions, with a corresponding drop in challenge level. While this may be an advantage for authors that would like to move toward less puzzle-based IF games, it may be a detriment to the traditional puzzle-based form.

Refined Design

My refined design retained in-text object links with associated pop-up action menus.

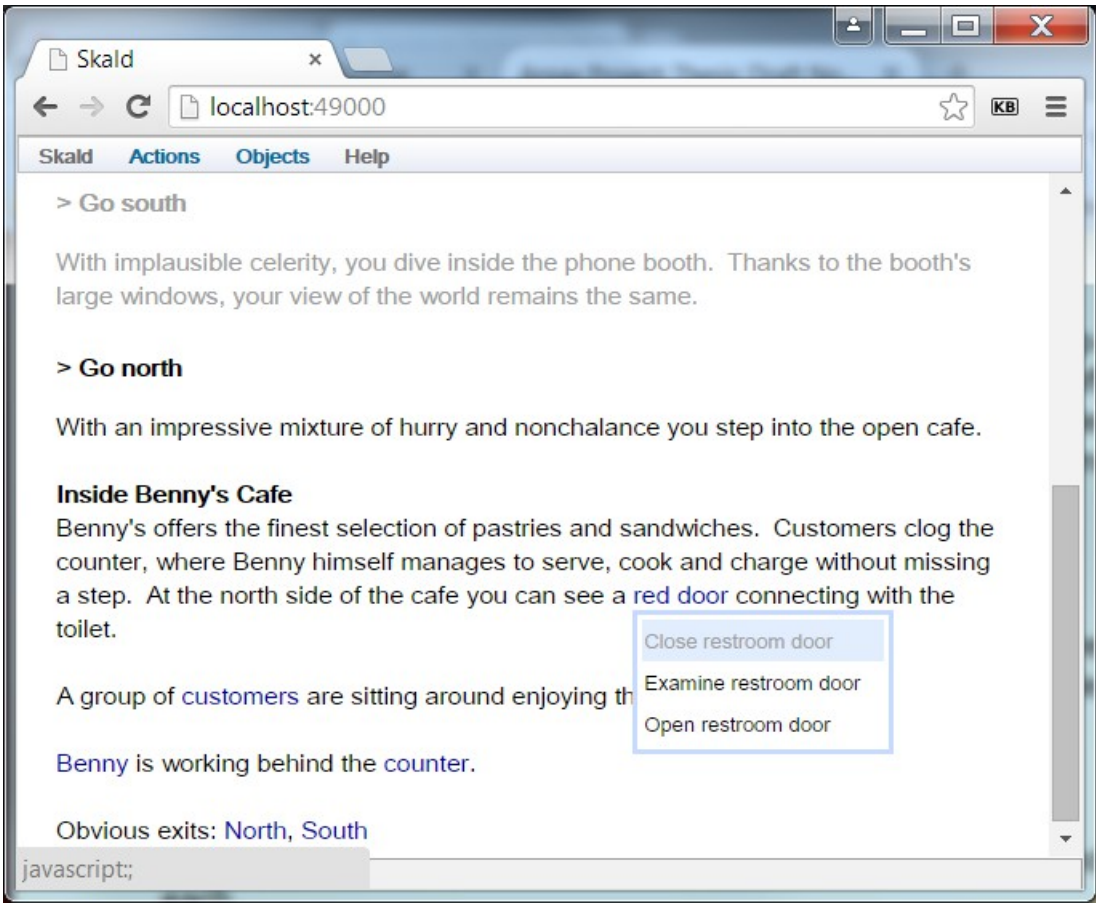

Figure 9: Skald interface, showing the pop-up menu resulting from clicking on red door

In future, I would like to provide users the option to control the formatting of those links, or even turn them off completely if they find them distracting. Also, one of the suggestions taken from the paper prototype tests but not yet implemented is that, when the user hovers over an object link, all other links to that same object should simultaneously turn the same color. This clarifies that these different links in fact refer to the same single object in the game world.

In order to clearly afford all possible actions at any time and to overcome the "interaction dead-end" problem, I supplemented the in-text object links with two menus.

The first menu lists all possible action verbs, with sub-menus for direct objects and possible indirect objects.

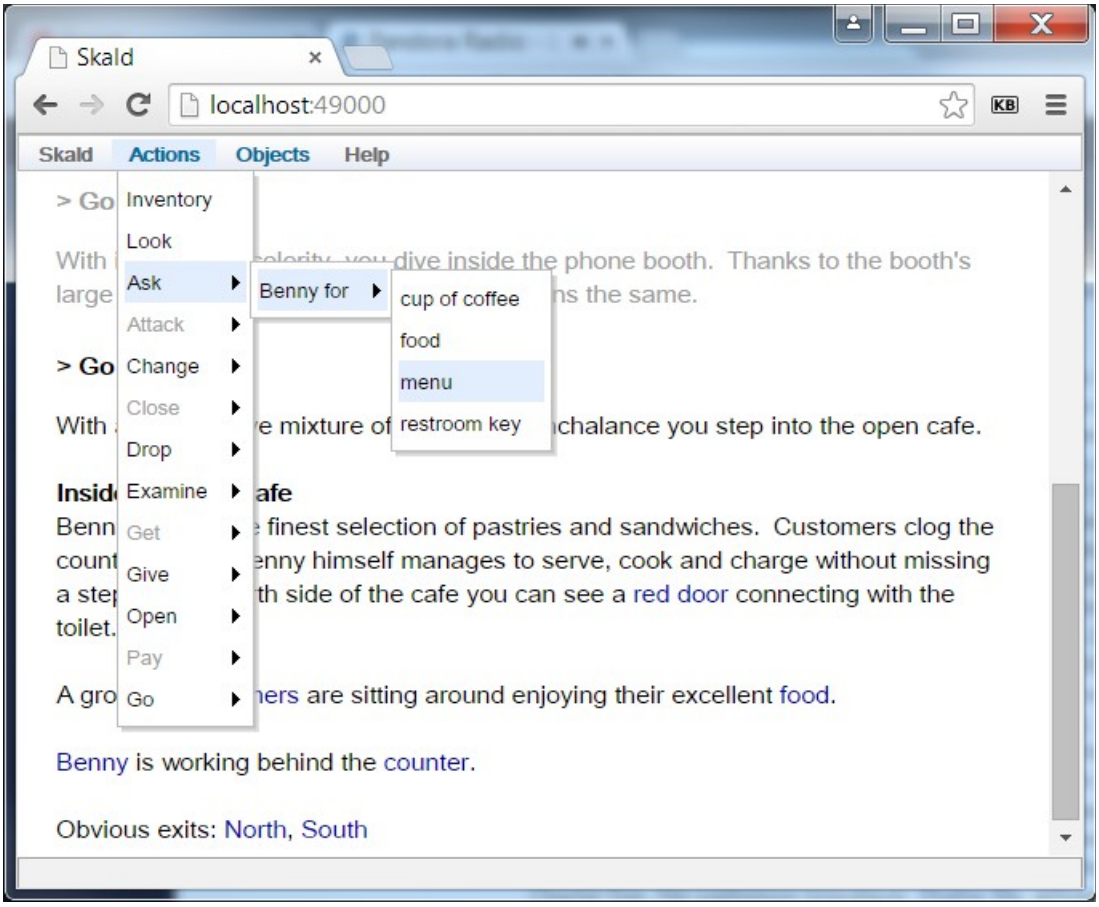

Figure 10: Skald interface, showing a sample Actions pull-down menu

The second menu gives a list of all currently-available objects, and then the set of actions that is currently possible to perform on each.

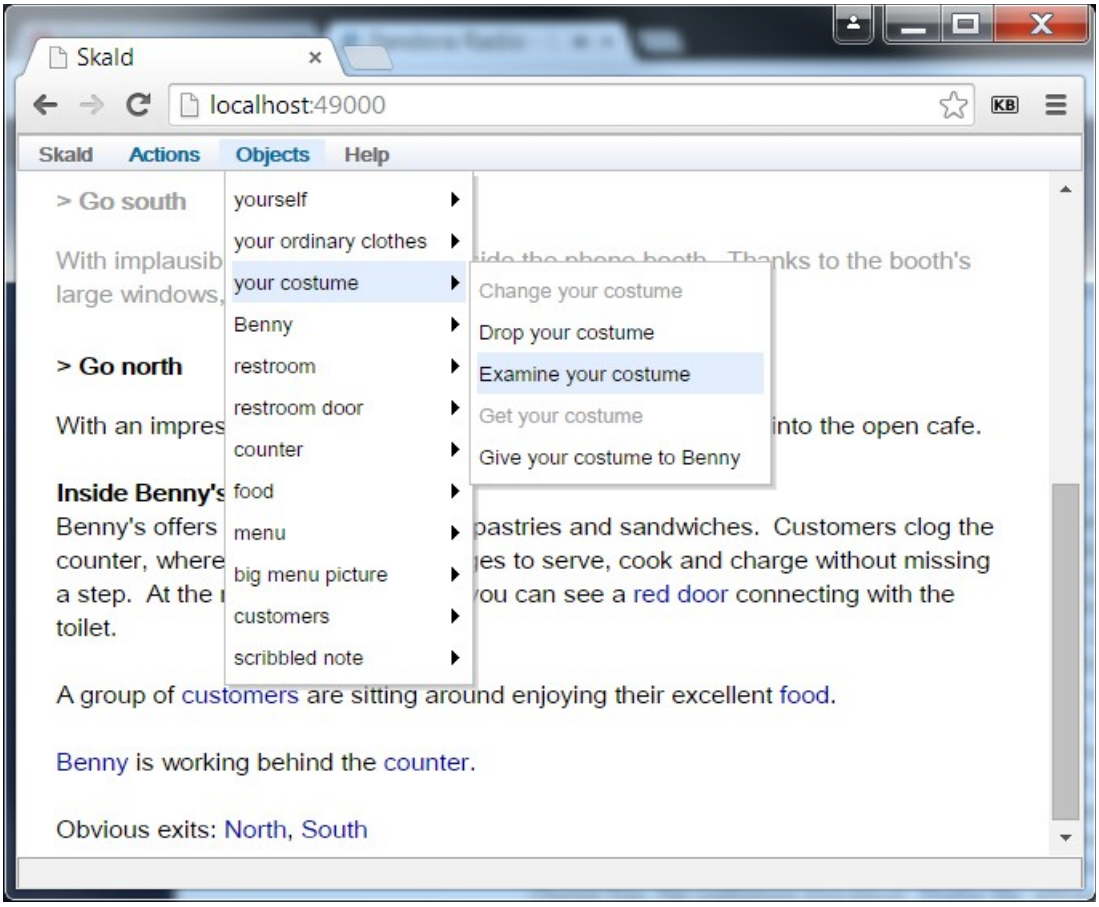

Figure 11: Skald interface, showing a sample Objects pull-down menu

To make these menus more obvious, it is also possible to view them as a left sidebar. This is the default start view in Skald. In this view, it is possible to switch between the Actions or Objects menus as tabs.

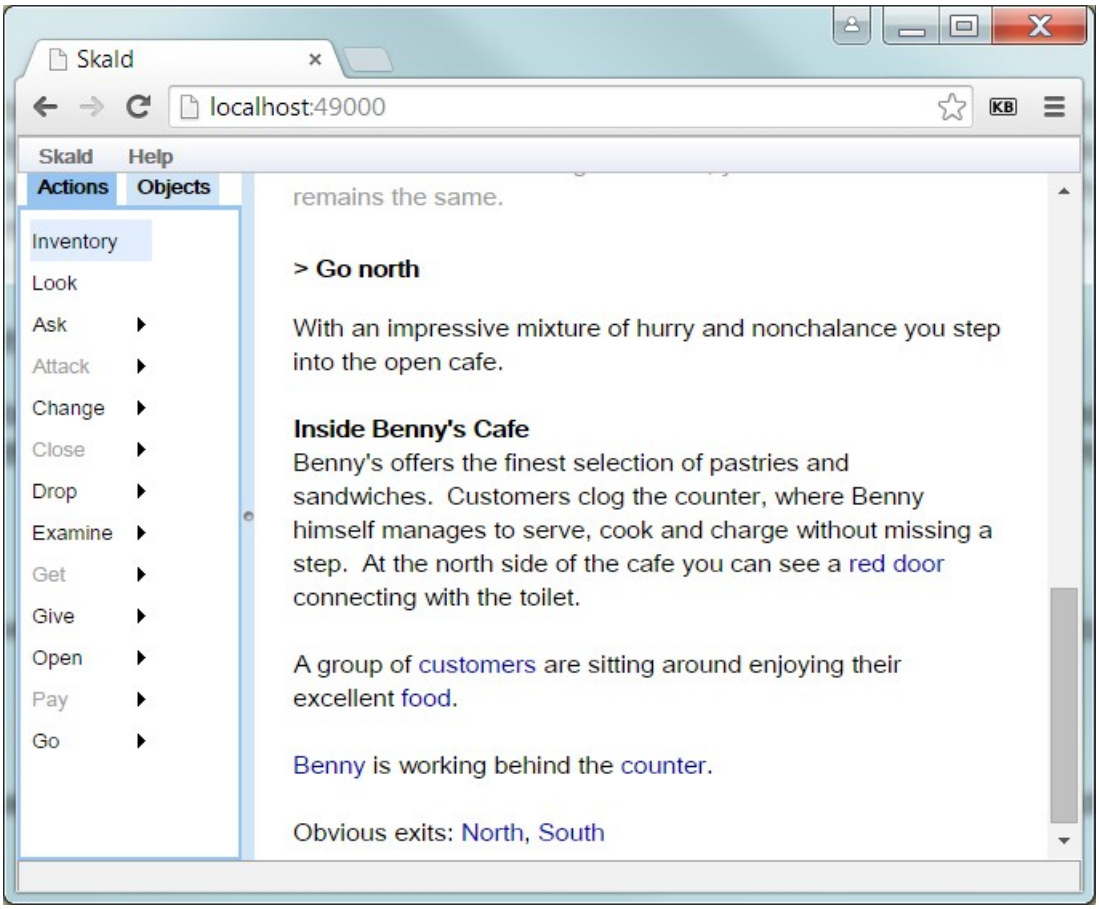

Figure 12: Skald interface, showing a sample sidebar menu

Other pull-down menus include an option to see the Skald system error log or a short *About Skald* message.

Implementation

In order to run in a browser and thus be accessible from both mobile and desktop devices, Skald is written using Google Web Toolkit (GWT). GWT provides a number of cross-platform UI widgets and a compiler that converts Java to cross-browser-compatible JavaScript. I chose GWT because I am comfortable coding in Java, but I do not know much JavaScript and have done little dynamic webpage development. GWT's compiler allowed me to write the Java code I'm comfortable with while shielding me from

most of the technical issues of dealing with different JavaScript incompatibilities across different browsers.

However, Skald only provides a user interface front-end. It does not model the game world. For that, I wanted to use one of the two most popular IF game engines: Inform or TADS. Inform is the more popular, and there are a number of projects that allow the entire Z-Machine or Glulx virtual machine (VM) to run in the browser itself. However, the user interface and back-end are quite closely linked in these virtual machines.

TADS, on the other hand, recently introduced a built-in web server to its VM. TADS then provides its own HTML and JavaScript-based user interface, but the server was designed such that these UI files could be swapped out with only a small amount of work. In addition, TADS has a more detailed system for parsing and handling user-entered actions than Inform. Each command must go through three phases: a verify phase to test how logical the command is, a check phase to see whether a logical command is in fact possible given the world state, and an action phase where the world state is actually affected. These separate phases made it easier to reuse the verify phase to generate the affordance lists—all of the logically afforded combinations of verb and objects—that Skald needs every turn to update its menus. The TADS programming language is also more robust and has a syntax similar to traditional programming languages. For the low-level work I needed to do for Skald, TADS seemed like the better starting point than Inform.

When a TADS game uses the Skald interface, it performs as follows. When the game starts up, it also starts the TADS internal web server. The server collects the initial game text, which consists of all the output before the first prompt for input.

Then, the server computes the current set of affordances. It does this by using an author-provided list of the verbs that might be used somewhere in this particular game. For each verb on the list, the server reuses the TADS parser's normal verify routines to poll each of the objects currently accessible to the player character to see if it presently supports that verb. For verbs that take two objects, it checks all combinations of currently-present objects for successful matches. The full list of all currently-afforded actions is then formatted into JavaScript Object Notation (JSON).

The server can then send to the player's browser the initial HTML and JavaScript code needed to display the Skald UI, followed by the initial text of the game and the first turn's JSON affordances data that Skald uses to build its menus. The links for each object in the game text are provided by the game author, who provides a unique name that is used by the system as an identifier to match the corresponding object in the JSON affordance data.

On each subsequent turn, the user's selected command is sent back to the server as an HTTP POST request. The server then replies with the resulting game text and updated affordance JSON data.

This client-server architecture means that Skald could be use as a front-end to other IF systems. All that is required is an HTTP server that can send game text and JSON affordance data in the format required by Skald and reply to HTTP commands as sent by Skald. Generating the necessary affordance data will generally require an intimate view of the underlying world model, however. This will generally require a custom plugin for each backing IF system, although this plugin could then be reused for any game built using that same IF system.

The Skald code and TADS plugin is available online (Tomaszewski 2013).

Potential Consequences

By doing away with IF's natural language parser, Skald has modified one of the key features of the medium. While it is hoped that this change will eliminate most user error and improve the clarity of what is currently possible in the game, this is not without side-effects. Without the command line prompt, there is no longer an illusion that any input might be supported. The boundaries of the game thus become clear, potentially reducing the challenge and eliminating the open exploration that provides some of the joy of the IF form.

An evaluation with human users was necessary to determine the significance of these changes.

EVALUATION

Hypotheses

I expect that, when compared to a traditional command line IF user interface, the menu-driven Skald user interface will improve the ability of users (particularly novice users) to play interactive fiction. Specifically, Skald will:

- Eliminate syntactically-invalid user input[5]
- Make it clearer to users which game-world objects currently support interaction
- More clearly afford which actions each of those interactive object currently supports

Together, these effects should increase users' sense of agency when it comes to moving through an IF game world.

This change in interface may also effect how people play the game in other ways. Examples include:

- playing faster or slower
- entering more or fewer commands per game
- interacting with a wider range of objects
- using a wider range of verbs (disregarding alternate synonyms for the same verb)

At a higher level, the Skald interface may reduce some of the engagement that comes composing commands without any prompting. Some users may find it to be a more tedious way to construct a command. Skald may also make certain puzzles easier to overcome, which would also reduce the challenge of the game, and thus possibly make the game boring. Alternatively, affording these actions may reduce frustration, thereby increasing engagement. Thus, I expect that Skald will affect users' engagement and enjoyment of IF, but not necessarily in a positive direction.

Experimental Setup

Subjects were recruited by email (Appendix A) sent to a number of student mailing lists at the University of Hawaii at Manoa. After completing an informed consent form (Appendix B) and background survey (Appendix C), participants were randomly assigned to two groups: Group CS and Group SC.

5 This is not the same as eliminating all user error. Users may still accidentally select the wrong menu option or otherwise make poor action choices.

Participants were then asked to play two different online text-based IF games: *Captain Fate* and *The Queen's Heart*. The games are described in more detail below. The order of the two games was the same for all participants, but the user interface of the two games varied. Group CS played *Captain Fate* with a traditional command line interface (CLI) and then *The Queen's Heart* using the new menu-driven Skald interface. Group SC first played *Captain Fate* using Skald and then *The Queen's Heart* using the command line interface. User input for each game session was recorded, which allowed for the generation of various game metrics, such as time spent and the range of verbs and objects used.

After each game session, participants were asked to complete a short one-page response survey (Appendix D). Finally, participants completed a one-page survey asking them to compare their two game experiences (Appendix E).

The Games

Spoiler warning: This section describes the primary puzzles and possible endings of the two games. You can skip this section if you would rather not know the details of the games before you play them. Knowing the details here is not necessary for you to understand the experimental results reported in the next chapter.

Captain Fate

Captain Fate was a humorous example game used in the *Inform Beginner's Guide* (Firth 2004). It was then converted to a TADS game and extended by Eric Eve (2005). With Eric's permission, I returned it to its original Inform length and modified it slightly to work with Skald.

The game opens as follows:

```
Impersonating mild mannered John Covarth, assistant help boy at an
insignificant drugstore, you suddenly stop when your acute hearing deciphers
a stray radio call from the police.  There's some madman attacking the
population in Granary Park!  You must change into your Captain Fate costume
fast...!

--------------------------------------------------------------------------------

Captain Fate
by Roger Firth and Sonja Kesserich
Converted to TADS3 by Eric Eve
Modified by Zach Tomaszewski

[Type HELP for instructions on how to play.]

--------------------------------------------------------------------------------
```

```
On the street
On one side - which your heightened sense of direction indicates is north -
there's an open cafe now serving lunch.  To the south, you can see a phone
booth.

>
```

As stated in the intro text, the goal of this game is to change into your superhero costume. It turns out that the inside of the phone booth is too visible to passers-by. That leaves the cafe to the north. Benny's Cafe has a restroom you can use, but you need to be a customer to ask for the key. So you have to order a coffee first. You have no money on you, but, if you are observant, you will find a lost coin in the restroom. Once you have paid for your coffee and returned the restroom key, you can head back out of the cafe to save the world.

Captain Fate has three possible endings, identified here by the ACTION that causes that ending. First, if you fail to lock the restroom door behind you, you will be interrupted while changing into your costume, thus revealing your secret identity (CHANGE). Second, you may be knocked unconscious if you try to attack Benny once you're in your costume (ATTACK). And finally you may successfully leave the diner in your superhero costume (SOUTH).

The Queen's Heart

The Queen's Heart is a game I authored as part of the 3-day Global Game Jam 2013 (Tomaszewski 2013). I then cleaned it up a bit and modified it slightly to work with Skald.

The game opens as follows:

```
You stand alone in the dark, sorrow in your heart and earthworms in your hand.

You stand listening.

You can hear a faint scratching... a sort of scurrying...  No, it is a
burrowing...  Yes, it is definitely a burrowing sound... though a rather small
one.  Too small for a badger, too steady for a rabbit, which leaves only one
thing: a mole.  There is a mole burrowing through the soft earth to your
right.

You look down at the earthworms in your hand.  "A bit of mole-fishing, then?"
you ask yourself.  "A fresh mole for the Queen, my love, to grant her the
strength she needs?"  You feel the warmth of a blush on your face, a rush of
blood.  You only call the queen "your love" in the most secret and lonely of
places, like down here in the tunnels.
```

```
You glance around carefully, first down the tunnel one way and then the other.
No one to see you, and no one to hear.  No one else in a long long time.

You see dimly but well enough in the dark, though you could walk and hunt
through these tunnels blindfolded if you wanted to.  You've guarded them for
so long that you know every twist and turn, every bump of moist soil.  As the
last of the Goblin Guard, this is your domain: The Warren of the Eternal Queen
(may she protect us all).

The earthworms are wriggling, and they bring you back to the task at hand.
"Right then.  Worms or mole for dinner?"

No one answers.

--------------------------------------------------------------------------------

The Queen's Heart
by Zach Tomaszewski

[Type HELP for instructions on how to play.]

--------------------------------------------------------------------------------

The Eastern Tunnels
Twisting tunnels stretch off in all directions through the raw earth.  Here
and there, stone arches and wooden support beams shore up the weight of the
soil above.  The soil is a little moister here than to the west.

A mole is poking the tip of his nose out of a small hole.

>
```

In this game, the player directs a lonely goblin knight who is guarding his sleeping Queen. His long solitude has left the goblin rather quirky, and he is not a very reliable narrator.

To the east is a large pool of water at the base of a well. To the west through the tunnels lies the Queen's Chamber. There, the Queen lies slumbering on her stone bier covered by a thick fur blanket. As the goblin wanders the tunnels, he has a recurring compulsion to check on the Queen to see if she is still alive. There are signs that he has been performing this same ritual over and over for years: holding his breath and listening for the Queen's soft heartbeat.

The initial mole puzzle has no effect on the final outcome, but it sets the tone of the game. The player can ignore the mole, lure the mole out of the hole and kill it, or simply let the mole have the worms. Killing the mole increases the sorrow that the goblin carries; feeding it lightens his burden.

Once the player listens to the Queen's Heart, there is a crash and a splash due to a little human girl falling through the rotted wooden cover of the well. The goblin then has the choice to push a large rock down onto the girl (KILL) or to climb up out of the well (UP). Once out of the well, the "iron taint of Man" removes the goblin's glammer so that he can be seen by the girl. The player can either choose to push a wooden plank down for the girl to climb up, or else he dislodges the plank when climbing back down for the same effect. The girl climbs out of the well and escapes. As with the mole, releasing the girl lightens the goblin sorrow. However, an observant player might notice that after a trip up to the outer world, the goblin sees things differently, as if with fresh eyes. For example, his once shiny sword is now tarnished with age.

Compelled to check on the Queen again, two possible endings are now possible. If the goblin killed the girl (KILL), he is protected from the outer world and his Queen still lives. On the other hand, if the goblin entered the outer world (UP), either his glammer is gone or his delusions were broken. When he listens to the Queen now, he fails to hold his breath first and finds that he cannot hear her heartbeat. Trying the ritual again, he realizes that perhaps the heartbeat he has been listening to all of these years is his own, beating in his ears. He is now ultimately alone and without purpose.

RESULTS

Participation Rates

The following table shows participant response rates and retention through the stages of the study.

Stage	Total Participants	Group CS	Group SC	Mean Time to Complete (minutes)
Received Email	250 to 500 *(estimated)*			
Number of clicks to the start of study	89			
Completed consent form	43	19	24	
Completed background survey	**41**	**19**	**22**	**4.6**
Opened first game (*Captain Fate*)	41	19	22	
Entered one or more commands	38	18	20	
Completed first game	**31**	**15**	**16**	**9.1**
Completed first response survey	**26**	**12**	**14**	**3.7**
Opened second game (*The Queen's Heart*)	26	12	14	
Entered one or more commands	23	12	11	
Completed second game	**23**	**12**	**11**	**13.1**
Completed second response survey	**23**	**12**	**11**	**3.4**
Completed comparison survey	**23**	**12**	**11**	**1.7**
Usable responses	**20**	**11**	**9**	

Table 1: Participant retention rates

Retention rates between the two groups were generally comparable. Once participants started the study by completing the consent form, the greatest drop in participants was while playing through the first game, regardless of user interface. It is also interesting that all three of the participants that aborted the study upon seeing the second game were in Group SC. They were transitioning from Skald to the command-line interface (CLI).

Of the 23 complete study responses, three were from players who reported that they had already completed the study or played the games before. These "repeated play" responses were dropped, leaving 20 usable responses. 11 of these final responses were in Group CS, and 9 were in Group SC.

Game Endings

As described previously, there are three possible endings to *Captain Fate* and two possible endings to *The Queen's Heart*. A few players also quit the game early, although they continued the study by completing the subsequent survey.

	CLI	**Skald**
Fate	6 CHANGE, 1 SOUTH, 4 QUIT	4 CHANGE, 2 ATTACK, 3 SOUTH
Queen	1 KILL, 7 UP, 1 QUIT	4 KILL, 6 UP, 1 QUIT

Table 2: Different game endings

Over a third of those who played their first game using the CLI quit the game early, while all of those who started with the Skald interface completed the game. Also, for both games, use of the Skald UI led to a wider variety of possible endings.

Participant Backgrounds

As mentioned, participants were asked to provide certain background information, such as age, gender, education level, and previous computer game experience (Appendix C). Thanks to random assignment of the participants to two groups, the groups did not significantly differ on most of these background measures, with two noticeable exceptions.

First, on a 1 to 5 scale, Group SC rated themselves as more familiar with a command line interface (Group SC mean: 2.78, Group CS mean: 1.73, $t(19) = 2.70$, $p = .11$). Secondly, Group CS rated themselves more familiar with the interactive fiction game genre (Group CS mean: 2.36, Group SC mean: 1.33, $t(19) = 3.72$, $p = .06$). These differences were on the border of statistical significance. However, given their high

potential to impact a new IF experience, the effect of both command line experience and IF experience will be examined in more detail below.

Game Experiences

Both survey responses and recorded game metrics were used to measure the differences in game experience along the dimensions described below.

Original survey questions are given in the title for each table. All survey responses used the following scale: Strongly Agree (5), Agree (4), Neutral (3), Disagree (2), Strongly Disagree (1).

Object Affordances

	CLI	Skald	Game Mean
Fate	2.55	3.56	3.00
Queen	2.89	4.27	3.65
UI Mean	2.7	3.95	

Table 3: It was clear which objects I could interact with in the game world.

When recognizing interactive objects, the difference between the CLI and Skald was statistically significant, $t(19) = 5.00$, $p < .0001$.

When comparing the two groups, there was also a significant effect between the games $F(1, 36) = 5.06$, $p = .03$, as well as the more significant interaction effect between the two interfaces, $F(1, 36) = 17.02$, $p = .0002$. This game effect suggests that game narration also plays a significant role at directing a user's attention towards interactive objects, regardless of the interface used.

Metric: Number of Unique Objects Used

Captain Fate contains 22 interactive objects: `Benny`, `Benny's Cafe`, `big menu picture`, `counter`, `cup of coffee`, `customers`, `food`, `light switch`, `menu`, `ordinary clothes`, `pedestrians`, `phone booth`, `restroom`, `restroom door`, `restroom key`, `scribbled note`, `sidewalk`, `silver coin`, `toilet`, `your costume`, `your ordinary clothes`, and `yourself`.

The Queen's Heart contains 27 interactive objects: `bier`, `blanket`, `depression`, `earthworms`, `flagstones`, `heartbeat`, `human child`, `iron cable`, `knot in`

`your stomach`, `large rock`, `mandala`, `(dead) mole`, `path`, `plank`, `poles`, `Queen`, `ragged hole`, `roots`, `shaft`, `small hole`, `sorrow`, `stone arches`, `(bronze/tarnished) sword`, `sunlight`, `support beams`, `water`, and `yourself`

Most participants only interacted with a subset of these per game.

	CLI	Skald	Game Mean
Fate	7.7	12.9	10.1
Queen	8.7	17.9	15.7
UI Mean	8.2	15.7	

Table 4: Mean number of unique direct objects interacted with per player

For both games, the use of Skald UI significantly increased the number of different objects that players interacted with, $t(19) > 3.9$, $p < .001$. The mean number of unique objects used nearly doubled when using Skald over the CLI. On average, this meant users engaged with over half of the available objects in both games, up from about one third of the available objects when using the CLI.

Metric: Degree of Object Use

Not only were more unique objects used with the Skald interface, but they were used to a greater degree. For *Captain Fate*, the following objects were used significantly ($p < 0.05$) more frequently as direct objects in commands under the Skald interface than when using the command line interface: `Benny's Cafe`, `restroom`, `restroom door`, `scribbled note`, `sidewalk`, `silver coin`, `toilet`, and `yourself`. No objects were used significantly more with the CLI.

For *The Queen's Heart*, the following were used significantly ($p < 0.05$) more often as direct objects using the Skald interface: `blanket`, `depression`, `heartbeat`, `knot in stomach`, `mandala`, `ragged hole`, `roots`, `small hole`, `stone arches`, `support beams`, `water`, and `yourself`. With the CLI interface, `Queen` was used significantly more often. (The most common text command related to the Queen is `listen to queen`, which the game actually maps to "*listen to heartbeat*". Under Skald, more people simply used the `heartbeat` object directly.)

In general, the objects used significantly more often with Skald tend to background objects or objects that are only mentioned subtly or passingly in the narration. None of them (except for Captain Fate's `silver coin`) needed to be interacted with in order to

finish the game. In other words, Skald seems to encourage more exploration and experimentation with a wider range of objects.

Action Affordances

	CLI	Skald	Game Mean
Fate	2.55	3.89	3.15
Queen	3.22	4.36	3.85
UI Mean	2.85	4.15	

Table 5: I knew which actions were possible to perform in the game

When determining possible actions, the difference between the CLI and Skald interfaces was statistically significant, $t(19) = 4.1$, $p < .001$.

Once again, when comparing the two groups, there was a significant effect between the games $F(1, 36) = 4.44$, $p = .04$, as well as the more significant interaction effect between the two interfaces, $F(1, 36) = 13.85$, $p < .001$. It seems that the nature of the game itself affects the range of possible actions that suggest themselves.

Metric: Number of Unique Verbs Used

Captain Fate contains 17 author-supported verbs: `Ask_for`, `Attack`, `Change`, `Close`, `Drink`, `Drop`, `Examine`, `Flip`, `Get`, `Give_to`, `Inventory`, `Lock_with`, `Look`, `Open`, `Pay`, `Travel`, and `Unlock_with`.

The Queen's Heart contains 14 author-supported verbs: `Attack`, `Climb`, `Drop`, `Eat`, `Examine`, `Get`, `Inventory`, `Kill_with`, `Listen to`, `Look`, `Push`, `Trace`, `Travel`, and `Wait`.

The command line UI also supports the `Help` verb, and some other library-provided verbs not relevant to completing the game. Use of these extra verbs was ignored here. The CLI also supports different synonyms that map to the same verbs listed above. For example, `Take` maps to `Get`. The following counts are based on the back-end verb used, irrespective of any different synonyms used to invoke it.

	CLI	Skald
Fate	10.1	13.3
Queen	9.4	9.6

Table 6: Mean number of unique author-supported verbs used per player

The effect of UI on was not significant for *The Queen's Heart*. However, the Skald UI did lead to a significantly broader use of unique verbs per user for *Captain Fate*, $t(9) = 3.4$, $p < 0.01$.

Metric: Degree of Verb Use

The two interfaces also prompted significant differences ($p < .05$) in the frequency with which the different verbs were used.

The most striking and significant ($p < .001$) effect is that the `Examine` verb was used over 3 times as often with the Skald interface for both games.

For *Captain Fate*, `Examine`, `Lock_with`, and `Open` were used significantly more often under Skald. These verbs are related to one of the important puzzles in this game—getting into the restroom. Forgetting to lock the restroom door before changing your costume leads to one of the less satisfying endings of the game. Affording this action more clearly seems to have reminded users to lock the door.

On the other hand, `Travel` was used more often with the CLI. Unnecessary travel between locations is often an indication at the player is at a loss for what to do next and is searching for clues.

For *The Queen's Heart*, `Examine` and `Kill_with` were used significantly more frequently with the Skald interface. `Attack` was used more significantly frequently with the CLI. This is an interesting shift in use between the interfaces: `Attack` and `Kill_with` both accomplish similar goals, though the `Kill_with` verb requires a second object use to use as a weapon—such as the goblin protagonist's bronze sword. So it seems that Skald may encourage users to try more complicated verbs and command structures.

Command Construction

	CLI	Skald	Game Mean
Fate	3.45	4.22	3.8
Queen	3.67	4.64	4.2
UI Mean	3.55	4.45	

Table 7: I was able to able to construct commands that the game understood

When it comes to the reported ease of constructing valid commands, the difference between the CLI and Skald interfaces was statistically significant, $t(19) = 3.9$, $p < .001$. However, the difference between the games was not statistically significant.

It is interesting to note that responses to this question are an average of 0.5 points higher than those to the questions regarding object and action affordances. This suggests that command construction is generally deemed easier than determining the objects and actions that can be used to construct those commands.

Metric: Total Inputs Entered

Counting total inputs involves some subtlety. An input corresponds to a successful command only with the Skald interface. For the traditional command line interface, the number of entered lines may be greater than the number of successful commands.

For example, some entered lines are syntactically invalid commands that result in an error message:

```
>door
The story doesn't understand that command.
```

Other inputs may contain misspellings. These also result in an error message, but the player can correct the command with a second line of input:

```
>examine soorow
The word "soorow" is not necessary in this story.

(If this was an accidental misspelling, you can correct it by typing OOPS
followed by the corrected word now.  Any time the story points out an unknown
word, you can correct a misspelling using OOPS as your next command.)

>oops sorrow
Sometimes your sorrow is a dark void in your chest, sometimes it is a gray
weight between your shoulders, and sometimes it is a languid emptiness
```

```
everywhere.  At the moment, it is very heavy.
```

Similarly, if a command is syntactically incomplete, it may take two lines of input to complete the command:

```
>examine
What do you want to examine?

>benny
A deceptively fat man of uncanny agility, Benny entertains his customers
crushing coconuts against his forehead when the mood strikes him.
```

The following table shows the mean number of inputs entered per game session for each experimental group.

	CLI	**Skald**	**Game Mean**
Fate	50.4	41.2	46.3
Queen	54.7	48.4	51.2
UI Mean	52.3	45.2	

Table 8: Mean number of total game inputs entered per game session

While, on average, players entered slightly more inputs when using the command line interface or when playing *The Queen's Heart*, neither the difference between the UIs or between the games was statistically significant.

Metric: Command Error Rate

For this metric, any command that does not affect the game state in an author-supported way is counted as an error.

As discussed above, an input may be syntactically invalid:

```
>door
The story doesn't understand that command.
```

An input may be syntactically valid command, but not refer to any present objects:

```
>give mole to queen
You see no queen here.

>buy coffee
You have no money.
```

A command may be syntactically valid and refer to existing objects, yet still be refused. This refusal may be a generic refusal as provided by the TADS library:

```
>attack customer
You cannot attack those.
```

Or it may be a refusal specifically written by the game author:

```
>talk to customer
As John Covarth, you attract less interest than Benny's food.

>ask benny for sandwich
Food will take too much time, and you must change now!
```

Because of the default behaviors provided by the TADS library, some commands may succeed and affect the state of the game world unless specifically prevented by the game author. For example:

```
>sit
(on the floor)
Okay, you're now sitting on the floor.

>stand
Okay, you're now standing.
```

However, since sitting or standing does not help with any puzzle or in any other way advance either of the games' stories, these commands still achieve nothing for these particular games. Therefore, as verbs not supported by the game author, they are counted as errors here.

	CLI	**Skald**	**Game Mean**
Fate	28.9%	0.3%	17.4%
Queen	20.8%	0%	10.0%
UI Mean	25.1%	0.1%	

Table 9: Percent of total inputs that did not produce an author-supported command

Due to a program bug, a single command entered using Skald failed to produce a valid command.

For the CLI, 25.1% of lines entered could be considered errors. If we disregard the commands that were simply not author-supported (but were still successfully processed by the default library) and look at only inputs that produced some sort of error message, this is still around 20.8% of inputs.

These error rate measures are in aggregate across all users. However, users who generate high input error rates tend to play shorter games and enter fewer inputs overall. If we computed a mean error rate by individual user, the mean user error rate over both games when using the CLI is 36.4%!

Every user of the command line UI produced at least one error or ineffective command.

Metric: Time Spent

	CLI	Skald	Game Mean
Fate	8.9	9.4	9.1
Queen	15.2	11.2	13.0
UI Mean	11.7	10.4	

Table 10: Mean time spent playing each game, in minutes

Across both groups, the maximum time spent on a game was 23.5 minutes and the minimum was 4.5 minutes. The difference between the two games was statistically significant, $t(19) = 3.20$, $p = .004$, but the differences between the two interfaces was not.

Metric: Command Input Speed

	CLI	Skald	Game Mean
Fate	5.85	4.85	5.4
Queen	4.23	4.53	4.4
UI Mean	5.1	4.6	

Table 11: Mean speed of input entry, in inputs per minute

Given the significant difference in time spent playing the two different games, the difference in game speeds was also significant, $t(19) = 2.47$, $p = .02$. However, the differences between the two interfaces was not significant.

Within both Group CS and Group SC, the fastest player using the CLI was also the fastest player when using Skald.

World-Level Agency

	CLI	Skald	Game Mean
Fate	3.45	3.78	3.6
Queen	3.67	4.09	3.9
UI Mean	3.55	3.95	

Table 12: I was sufficiently able to direct my character's actions in the game world—such as moving from place to place, manipulating objects, talking to other characters, etc.

While Skald was rated slightly higher regarding user's sense of agency within the game world, the difference was not significantly significant.

Story-Level Objectives

	CLI	Skald	Game Mean
Fate	3.73	3.56	3.65
Queen	2.67	2.55	2.6
UI Mean	3.25	3	

Table 13: I usually knew what I was expected to do in the game (even if sometimes I had to figure out exactly how to accomplish it)

Regarding story-level direction, there was a significant difference between the two game means, $t(19) = 3.9$, $p < .001$, but not between the user interfaces.

As proposed by the poetics for interactive narrative discussed previously (Tomaszewski & Binsted 2006), a story's content should provide narrative constraints or similar implicit suggestions regarding the appropriate world-level user actions. These results were a nice confirmation that the story does indeed make a difference to this experience.

Captain Fate is a classic superhero scenario. The first paragraph explicitly lays out the player's objective: find a place to change into his costume. It was fairly obvious what the player was supposed to accomplish; the challenge lay in figuring out how to achieve it.

On the other hand, *The Queen's Heart* opens on an unconventional scene of a lonely goblin living in a dark tunnel. No end-game objective is given, only a suggestion of what the first action might be regarding a nearby mole. Interestingly, as shown above, the

object and action affordances were still clearer in *The Queen's Heart*. So players could more easily find things to do in this game, but they didn't know what they were *supposed* to do.

Interactive Story Experience

	CLI	Skald	Game Mean
Fate	4.00	4.00	4.0
Queen	4.00	4.18	4.1
UI Mean	4.0	4.1	

Table 14: The game session had a story-like structure

	CLI	Skald	Game Mean
Fate	3.73	3.44	3.6
Queen	3.78	3.73	3.75
UI Mean	3.75	3.6	

Table 15: The game contained puzzles or situations that required some thought to overcome

	CLI	Skald	Game Mean
Fate	4.09	3.56	3.85
Queen	3.78	4.18	4.0
UI Mean	3.95	3.9	

Table 16: I believe that the game may have had a different outcome had I performed different actions

These three questions explore the essential features of a classic interactive fiction experience: that it has at least some narrative structure, that it includes challenges and puzzles for the player to work through, and that there is some sense of significant choice on the part of the player.

While each of these dimensions received a fairly favorable rating, there was no statistically significant difference based on the user interface used, game played, or the participants' groups.

Summary Experience

	CLI	Skald	Game Mean
Fate	3.00	3.67	3.3
Queen	3.44	3.73	3.6
UI Mean	3.2	3.7	

Table 17: I enjoyed playing this game

Despite the differences in the games and the user interfaces, neither had a significant effect on the overall enjoyment of the games.

Open-Ended Responses

Below are the answers given in response to three open-ended questions. These answers have been tagged with interesting themes discovered across multiple responses. This tagging is fairly subjective. It was performed by only a single tagger, and so it lacks any inter-rater reliability. Also, it was sometimes difficult to determine the full meaning of a short comment. When more than one interpretation was possible, tags were given followed by a question mark. Such "questionable tags" were only counted as a half-point in the summary totals.

However, despite these limitations, these tags still serve to highlight issues related to some of the aspects explored above as well as to raise some other interesting issues.

Least Enjoyable Aspect

The following are answers to the question, *What was the LEAST enjoyable aspect of this game?*.

Group CS - *Captain Fate*, CLI

- Unfortunately it ran very slowly, which made minutiae like unlock door, open door, close door, etc. rather tiresome
  UI behavior | actions too low-level
- trying to figure out how to do certain things
  unclear affordances
- Little explanation as what kind of commands I could use before i typed "help"
  unclear affordances

- some commands seemed less efficient than they could have been. GO, for example, instead of WALK.
  unclear affordances *(the game lacks a WALK synonym for the GO verb)*
- I wasn't entirely sure exactly what the goal of the entire game is (even now, since I lost). I don't know whether or not it is to get dressed into hero clothes or solve some other crime past the part where I lost. The description of the game needs to be elaborated a little bit more.
  unclear objective
- "Buy" was an infeasible command as I had no money, but Benny gave me coffee for free when I asked?
  actions too low-level *(you have to ask for coffee and then pay for it later once you find some money)*
- Solving the problem
  unclear objective*?* puzzles too challenging*?*
- I had to consult the help menu for commands multiple times. Simple commands like "use key" at the door didn't work, and having to type "use key on door" seemed very cumbersome with a such a finite number of objects and small game world. Also, since the door required a key to enter, it didn't occur to me I needed to lock the door manually, so that version of defeat seemed antiquated.
  unclear affordances actions too low-level
- I couldn't figure out how to get money to pay benny with, and therefore couldn't get the key to unlock the bathroom door with. When I tried to restart, the same screen came up so I couldn't try going to the phone booth instead.
  puzzles too challenging UI behavior*? (intentional behavior for the evaluation session)*
- The text command-based game was a type I haven't played in a very long time. *?*
- The least enjoyable aspect of the game was the game's inability to adapt to basic requests/commands/movements/actions. I often received the response "The story doesn't understand that command". That's a response I expected, but it was frustrating to get that answer when I tried to act in ways that I thought were pretty obvious. I also skipped through some basic actions (walk into bathroom first requires getting the key, unlocking the bathroom door, opening the door, then walking in, closing the door, locking the door, etc.) to the point where it was just mundane.
  unclear affordances actions too low-level

Group SC - *Captain Fate*, Skald

- There was very little set up when first entering the game. It felt like I was presented with a screen and I started to randomly click to figure out what to do.
  unclear objective
- getting stuck in benny's cafe
  puzzles too challenging
- trying to figure out which commands would get me into the restroom
  unclear affordances*?* puzzles too challenging*?* actions too low-level*?*
- It took quite a bit of tinkering with the commands to figure out what I had to do, especially when looking for the money to pay Benny.
  unclear affordances*?* puzzles too challenging*?* actions too low-level*?*
- The movement aspect. Why not just say go "cafe" or go "phone booth" instead of "north" or "south". Was kinda confusing at first.
  unclear affordances
- It was frustrating that links in text prompts became unclickable as soon as another prompt appeared, although that drove me to focus on the menu.
  UI behavior
- The suspicion that I had no freedom whatsoever and was merely being led through a predetermined set of moves.
  limited or unclear story paths
- no branching paths, couldn't do stupid shit to fail
  limited or unclear story paths
- Some actions were hard to discover (reading the note on the bathroom door for example).
  unclear affordances

Group CS - *The Queen's Heart*, Skald

- the greyed-out options!
  UI behavior
- trying to figure out how to do certain things
  unclear affordances*?* puzzles too challenging*?*
- Too many options to click on
  UI behavior*?* unclear (excessive) affordances*?* disliked click input*?*
- I would have liked to enter in my own commands (like the first game), but with the list of actions on the sidebar (like the second game). It just felt like I was clicking things to see what I could do because I was, but I'm not sure if that was important or not.
  disliked click input

- The game is entirely based on clicking. I can't type anything, which makes the game slower and more confusing (since there's already a pre-set choice of actions). I didn't know exactly what to do, or even what the goal of the entire game is again.
  disliked click input | unclear objective
- Lack of objectives.
  unclear objective
- The end when the queen died
  story event
- Then queen died pretty suddenly and without really any explanation. I had no idea what I was supposed to do differently in the game so that she didn't die at that point. The game felt incomplete.
  story event | limited or unclear story paths
- I wish I hadn't killed the little girl. I wonder what would have happened if I didn't push that rock on her.
  story event
- I wasn't sure of what I needed to do next or if I hit a dead end.
  unclear objective
- I wish there was a Back button!
  UI behavior

Group SC - *The Queen's Heart*, CLI

- I had to type help all the time because i could not remember the specific action commands.
  unclear affordances
- I didnt know how to use the sword or save the queen :(
  unclear affordances | puzzles too challenging
- I had no idea what the objectives were, when to look, where to go. I just started typing commands to see what would happen
  unclear objective
- It was a good game, but not having played many games of this kind it was a little frustrating to figure out what to do.
  unclear objective
- Interaction aspect. Was not entirely sure what I could interact with or what command to initiate the interaction.
  unclear affordances
- not knowing what I could do, or what I needed to do
  unclear affordances | unclear objective

- Again, feeling like I was being led through a series of predetermined moves, i.e. that I was not free.
  limited or unclear story paths
- guessing randomly on what to do
  unclear objective unclear affordances*?*
- The list of commands was too small. It seemed like it was supposed to be more open, having the ability to free form commands. But, the list was too small so some actions were awkward to perform. For example, a USE or PERFORM command might have been nice.
  unclear affordances

The following is a summary of the tag counts for each experimental condition.

	Group CS: *Fate*, CLI	**Group SC: *Fate*, Skald**	**Group CS: *Queen*, Skald**	**Group SC: *Queen*, CLI**
unclear affordances	5	3	1	5.5
actions too low-level	4	1		
puzzles too challenging	1.5	2	0.5	1
unclear objective	1.5	1	3	4
story event			3	
limited or unclear story paths		2	1	1
UI behavior	1.5	1	2.5	
disliked click input			2.5	

Table 18: Tag summary for least enjoyable aspect of game

We can see here some of the same trends noted above with the qualitative measures. For example, more people complained about unclear affordances when using the CLI. There were more complaints about actions being too low-level in *Captain Fate*, while *The Queen's Heart* lacked a clear story objective.

It is interesting to note how many complaints there were about the click-based input from Group CS when playing *The Queen's Heart*. This is the group that gave the highest rating to object affordances, action affordances, command construction, world-

level agency, and enjoyment. So while they found the UI easier to use and seemed more immersed in the story, they still missed the ability to type commands.

Most Enjoyable Aspect

The following are answers to the question, *What was the MOST enjoyable aspect of this game?*

Group CS - *Captain Fate*, CLI

- the description of benny was pretty good
  narration
- actually figuring out how to do certain things
  puzzle challenges
- Trying to figure out exactly what I needed to do.
  puzzle challenges
- paying for my coffee :)
  puzzle challenges*?* progress/advancing the story*?*
- I enjoyed the originality of the game itself. Most of the time, it is out to get somebody, but this game requires changing clothes as the main goal (unless I'm wrong, since I lost and I don't know if it goes past this point).
  story
- Figuring out the situation. The game required thought, but the answers were generally available to the attentive.
  puzzle challenges
- The interaction
  interaction
- I liked going to the bathroom after Benny asked me to pay for the coffee; I was hoping I could stiff him as a superhero but unfortunately I was caught naked and the game ended.
  interaction *(roleplaying, interesting choices)*
- This is my first time playing this sort of game, so it was an interesting experience.
  game medium/genre *(novelty)*
- I liked how realistic the story is made, and how it went throughout the game.
  world simulation/verisimilitude*?*
- I really didn't enjoy the game.

Group SC - *Captain Fate*, Skald

- The descriptive text was very entertaining. I found it fun to be able to find an action to react to something I had just read.
  narration interaction
- clicking a lot of stuff
  interaction*?*
- it reminds me of a choose your own adventure book.
  interaction*?* story*?* game medium/genre*?*
- It was amusing to play the game.
  fun/humor
- The story aspect. The consequences of some actions were quite humorous.
  story fun/humor
- No wasted time guessing or experimenting with inactive or unusable nouns or verbs.
  clear affordances
- The end.
  progress/advancing the story*?*
- ui was novel and easy to use
  UI clear affordances
- The story was well done, and the actions that were expected did seem to unfold in a real way.
  story world simulation/verisimilitude*?*

Group CS - *The Queen's Heart*, Skald

- the story was nicely written
  narration
- accomplishing tasks
  puzzle challenges*?* progress/advancing the story*?*
- The mystery of now knowing what was going to happen with each command.
  interaction
- reading about the emotions the character carried
  narration
- I liked that there was another type of text-based gaming without the use of the keyboard.
  UI *(point-and-click)*
- Story: construction of the world and unfolding of events.
  story world simulation/verisimilitude*?* progress/advancing the story*?*

- The storytelling
  narration story
- I was ambivalent but intrigued by the fairly slow revelation of what you were and how magic interacted with you. When I learned I was a goblin, I in some ways wished I had learned earlier, but also recognized that was probably intentional so you didn't feel alienated from the character immediately. On the other hand, I did really wish I had more description of the character somehow, at least in terms of weight if climbing up and down the hole was going to change the environment.
  narration
- This game was a lot more manueverable than Captain Fate. It was easier to tell what I could do because all of my options were laid out before me.
  clear affordances
- I liked that it gives me possible actions I can perform instead of having to guess commands.
  clear affordances
- Being able to figure out what options were available to me
  clear affordances

Group SC - *The Queen's Heart*, CLI

- The subtle story behind the goblin was very interesting and very well integrated into the game. Whenever you got far from the queen the knot in your stomach description would change. I thought that was cool.
  story narration
- i figured out how to save the girl at least and get the mole :D
  puzzle challenges progress/advancing the story*?*
- Pushing the plank?
  interaction*?*
- It had a very unique story that I enjoyed playing.
  story *(novelty)*
- Puzzle aspect. Trying to figure out how to accomplish things was kind of fun.
  puzzle challenges
- melancholic and whimsical atmosphere
  narration
- The weirdness of the story.
  story*?* narration*?*
- more challenging than other game, could (and did) actually fail
  puzzle challenges interaction *(branching story paths)*
- I did still enjoy the ability to freely form the actions. In the previous game it was nice to have the set of all actions available in a menu style layout, but it seemed

more restrictive and contrived. This felt more real to me.
UI *(keyboard)*

The following is a summary of the tag counts for each experimental condition.

	Group CS: *Fate*, CLI	**Group SC: *Fate*, Skald**	**Group CS: *Queen*, Skald**	**Group SC: *Queen*, CLI**
UI		1	1	1
game medium/genre	1	0.5		
narration	1	1	4	2.5
world simulation/ verisimilitude	0.5	0.5	0.5	
clear affordances		2	3	
puzzle challenges	3.5		0.5	3
interaction	2	2	1	1.5
progress/ advancing the story	0.5	0.5	1	0.5
story	1	2.5	2	2.5
fun/humor		2		

Table 19: Tag summary for most enjoyable aspect of game

As the name *interactive fiction* suggests, this table highlights that the most enjoyable aspects include the text narration, the unfolding story, and ability to interact in that world. It is interesting to note that, while the Skald interface provided clear affordances, it was the CLI that made the puzzle challenges enjoyable. Also, the UI comments were fairly split—some liked the keyboard better and some liked the point-and-click better.

Comments

See Appendix F for participants' responses to the question, *Any other comment?*

User Interface Comparison

In the last step of the study, participants explicitly compared the two user interfaces on a number of different measures.

Question	Group	Response Counts					Mean
		CLI (definitely/ -2)	CLI (slightly/ -1)	No Prefer- ence	Skald (slightly/ +1)	Skald (definitely/ +2)	
Which interface did you find easier to use?	CS	2	0	0	2	7	1.09
	SC	2	1	1	1	4	0.44
	Total	**4**	**1**	**1**	**3**	**11**	**0.80**
Which interface did you find faster to use?	CS	1	1	0	2	7	1.18
	SC	4	0	1	2	2	-0.22
	Total	**5**	**1**	**1**	**4**	**9**	**0.55**
Which interface was more enjoyable to use?	CS	3	0	1	1	6	0.64
	SC	3	1	1	2	2	-0.11
	Total	**6**	**1**	**2**	**3**	**8**	**0.30**
Which interface made the game more interesting or engaging?	CS	2	0	1	2	6	0.91
	SC	3	3	1	0	2	-0.56
	Total	**5**	**3**	**2**	**2**	**8**	**0.25**
If you were to play a third game, which interface would you prefer to use?	CS	2	0	1	1	7	1.00
	SC	4	0	2	1	2	-0.33
	Total	**6**	**0**	**3**	**2**	**9**	**0.40**

Table 20: Responses to questions explicitly comparing Skald and the command line interface

Based on the mean of all responses for each question (shown in bold), participants slightly favored Skald over the CLI for all categories. The means range from 0.25 to 0.80 on a -2 (CLI) to 2 (Skald) scale.

However, there is much more going on here. First of all, the distribution of responses for every question are clustered at the two extremes of the scale, with few participants falling in the middle. So most people strongly favored one interface or the other, with slightly more favoring Skald over the traditional command-based UI.

Secondly, there was a noticeable difference between the two group means. Group SC favored the CLI side of scale for all question except ease of use (where they still had a lower score than Group CS). On the other hand, Group CS favored Skald over the CLI-based UI. This may just be an anomaly of group formation, or it may be that people favored the interface they played second.

Comments

Participants' additional comments can be found in Appendix G. These comments tended to underscore the qualitative results explored above:

- Some users clearly prefer the CLI using keyboard input while other clearly prefer Skald's menu-driven interface with point-and-click input.
- Those who prefer using the CLI appreciate the challenge and engagement it engenders.
- Those who prefer using the Skald UI appreciate the clear affordances it provides.
- This does not need to be a binary decision. One interface could offer both styles, where users can choose whether to explore the menus or to simply type commands.
- A game's user interface style is but one aspect of a user's total experience. The game's content provides significant clues to what the player can and "should" be doing.

Effect of Participant Background

As mentioned previously, there was an uneven distribution of prior experience between the randomly-assigned groups regarding using a command line and playing interactive fiction. Specifically, Group SC rated themselves as more familiar with a command line interface and Group CS rated themselves more familiar with the IF genre. It is therefore worth taking a closer look at the correlation between individuals' prior experience and their reported game experience.

In this analysis, a Pearson correlation score of $r >= |0.44|$ is considered statistically significant, equivalent to $p <= .05$.

Command line Experience

Across all participants, there was a significant correlation between reported command line experience and the participant's reported ability to construct commands (r = 0.64), agency within the game world (r = 0.52), and their enjoyment of the game overall (r = 0.63) while using the CLI. When using the Skald UI, there was no correlation with previous command line experience.

Assuming that Group SC's higher degree of command line experience may have inflated their ratings on these three dimensions when using the CLI does not endanger any of the conclusions drawn above. In fact, a slightly lower rating from Group SC on their enjoyment of the CLI may have lead to a statistically significant result regarding the effect of Skald on enjoyment.

There was a significant correlation between participants' familiarity with the command line and their preferences for the traditional IF command interface as measured by the final comparison survey. These correlated measures included rating the CLI as faster to use ($r = 0.53$), more enjoyable to use ($r = 0.45$), creating a more interesting or engaging game ($r = 0.63$), and preferring it as the UI in future games ($r = 0.52$). However, there was not a significant correlation with finding the CLI easier to use ($r = 0.26$).

Interactive Fiction Experience

Across all participants, those with prior interactive fiction experience reported finding it easier to recognize interactive objects ($r = 0.71$), determine possible actions supported by those objects ($r = 0.69$), construct commands ($r = 0.55$), and generally exhibit agency in the game world ($r = 0.46$), but only when using the Skald interface. Surprisingly, this prior experience showed no correlations at all with response ratings when using a traditional command-based IF user interface.

If we assume that Group CS may thus have rated Skald higher on these four measure due to their higher levels of prior IF experience than Group SC, it would bring into question some of the conclusions above regarding Skald's ease of use for all users. However, those conclusions were all very significant ($p < .001$) and so would probably not be unduly influenced by the slightly uneven distribution of IF experience between the two groups.

There was no correlation with previous IF experience and any preference for one UI over the other, as reported on any rating in the final comparison survey.

DISCUSSION

The results of this study imply the following conclusions.

IF is a daunting computer game genre. Faced with an interactive fiction game, about one third of participants failed to complete the first game, regardless of the UI used. Participant comments from those who did complete both games still indicated that IF is not everyone's cup of tea.

The traditional IF command line interface has a high input error rate. Across all textual inputs of all participants, approximately 25% of them failed to invoke an author-supported action. Approximately 20% of inputs actually produced an error message. Given that some frustrated users encountered even higher individual error rates, the mean user error is 36%. Every user of the command line interface produced at least one error or ineffective command.

However, participants' survey responses showed that this does not put everyone off of this user interface. Many still preferred the command line interface versus its Skald alternative.

Skald was easier to use than the traditional IF user interface. Skald eliminated the high number of inputs that did not map to author-supported actions. Object affordances, action affordances, and command construction were all rated as being significantly easier when using Skald. 70% of participants also rated Skald as slightly or definitely easier to use than the traditional command UI.

Skald encouraged players to explore the game world to a greater degree. This manifested in a few different ways. First, players used more game world objects. When using the Skald UI, a significantly wider range of objects were used at least once in an action by players. This was not only a one-time use, either; many of those objects were also used significantly more often. Most of the objects that saw additional use were background and scenery objects.

Second, a wider range of unique verbs was used—although this was a significant difference only when playing *Captain Fate*. Players are still selective, though, as no player ever used every verb, even with Skald.

Most significantly, the `Examine` verb got triple the use in both games when using the Skald UI. As a game author, this is rewarding: people are actually reading all the descriptions I authored! This closer examination of the world in particular—and the

wider use of verbs and objects in general—suggests that authors could potentially make clues to a game's puzzles more subtle when using the Skald UI.

Skald's effectiveness did not shorten participants' game sessions. Skald maps every user input to a valid, author-supported command. However, this did not significantly affect the number of inputs entered during a game, the time spent playing, or the speed of input entry. In short, players are still spending the same amount of time and making the same number of inputs with Skald, but, instead of generating errors 20% of the time, they are exploring game world objects to a greater degree.

Skald does not eliminate all affordance or Gulf of Execution problems. Even when using Skald, unclear affordances were mentioned by a number of participants as the least enjoyable part of their experience. However, this complaint was more often vaguely phrased as "figuring out what to do" was difficult, which could be interpreted as a problem of not having enough options afforded, or having too many options afforded, or having an unclear story objective to provide the player with clear goals and intentions.

Interestingly, in the spirit of IF as a riddle and a challenge, "figuring out what to do" was also given as one of the most enjoyable aspects by some other participants.

More than just material affordances go into determining an experience of world-level agency. The poetics theory of interactive drama states that a user will experience agency provided they have sufficient material affordances and the story-level constraints needed to inform their choices. Although Skald significantly improved participants' ability to recognize both interactive objects, possible actions, and construct commands, it did not significantly impact their feelings of world-level agency. Given that world-level agency did not correlate either positively or negatively with story-level objectives, it suggests that something more is going on here than this model suggests.

One likely possibility is that, if a perceived affordance is a relationship between the system and the player, then some feature of the player is also an essential part of this equation. This might be their confidence level, attention to detail, investment in the story, etc. The correlation found between the degree of previous CLI experience and the degree of reported world-level agency lends some credence to this. It seems that future work is required to clarify the full interaction between game world affordances, story level constraints, and the player's intrinsic traits.

The game itself makes a big difference. This is fairly obvious, but the game's content will generally have a greater effect on the player's overall experience than the UI. At the story level, a game is defined by the richness of the narration, complexity of the puzzles, structure of the story, and the depth of the non-player characters. Depending on the author, a game may be longer or shorter, containing more or fewer locations and

objects, with those objects supporting more or fewer interactions. While the user interface may make this content more or less accessible to a player, it cannot compensate for defects in this content itself.

In terms of the two specific games used here, there were statistically significant differences in the time required to play the game, user-reported object affordances, action affordances, and the clarity of story-level objectives, regardless of the UI used. There were also some UI-specific effects. For example, Skald lead to more unique verbs being used for *Captain Fate*, but not for *The Queen's Heart*.

The effect of UI on story-level concerns is subtle. The UI used did not affect most higher-level concerns, including users' reported sense of story-level objectives, experience of a story-like structure, challenges or puzzles that required some thought to overcome, or the presence of significant story-affecting user choices. While some trace of the UI seems evident in overall enjoyment, this effect was not statistically significant.

The UI does still have some subtle story-level influence, though. The use of Skald resulted in fewer participants quitting the game played (yet still continuing with the study), and it increased the distribution among possible story endings.

I had some concern that Skald would change how users play IF by making puzzles too easy. Results here suggest that it actually makes different story paths more obvious, allowing more players to avoid the more unpleasant possible endings.

Despite its advantages when it comes to ease of use, Skald is not a clear winner with all players. Although more players favored Skald, 30% still stated that they would strongly prefer the command line interface for future games. On other comparison questions—such as which UI they found faster, more enjoyable, and more engaging—opinion was similarly distributed to the extremes. Few participants were middle-of-the-road here. This aligns with previous research (Mehta 2010), which also found that, despite the recognized limitations of free-text input, usually around half of participants still prefer it for the sense of open-ended agency that it offers.

Prior CLI experience seems to be at least one factor this in preference. Familiarity with command line environments corresponded with the reported ability to more easily construct valid commands, exert world-level agency, experience engagement, enjoy the game overall, and prefer a command line interface for future games. Interestingly, prior experience with IF did not have a significant impact on which UI a player preferred, by any of the various measures used here (easy of use, speed, engaging, etc.).

CONCLUSION

A work of interactive fiction (IF) can be defined as an explicitly-modeled narrative world with a text-based user interface. Interactive fiction games were once a dominant commercial game form, and new games are still produced by a small but active community. IF has also been applied to various "serious" non-entertainment goals, such as education and interactive drama research.

Skald is a new interface for IF games. Skald is still text-based, both for output and user input. However, it is menu-driven, rather than offering free-form input on a command line.

As hypothesized, Skald has shown a number of interesting effects on game-play with real users. It successfully eliminates syntactically invalid user input, and it makes object and action affordances clearer—though not crystal clear for all players. Skald did not significantly change players' game speed or the time they spent playing a game, but it did lead to players using with a wider range of both objects and actions. In particular, they did much more examining of game world objects.

Skald was rated as easier to use than the traditional IF interface by 70% of participants in this study. However, this did not correspond to a significant increase in world-level agency, which suggests that certain features of the player also impact this feeling of agency. For example, it was found that previous command line interface experience corresponded with a higher sense of world-level agency, although previous IF experience did not.

Skald had no significant effect on users' reported experience of the story—the structure of the story, the challenge of the puzzles, the impact of their choices, or their overall enjoyment. However, Skald did have some subtle effects on the story structure, increasing the distribution of possible endings that player's reached.

Despite Skald's advantages and it being the preferred UI for a slight majority of users, the traditional IF interface was still preferred by a substantial minority of players.

Future Work

The findings here suggest at least two avenues for future work.

First, Skald should be expanded to also support a command line input alternative, possibly revisiting the "real-time parsing" design considered in Chapter 4. As some participants suggested in comments, a hybrid design may still be valuable. As with

Legend Entertainment's design, this may offer some blending of the best of both worlds, depending on player preferences. As currently architected, this will be a bit of a challenge, given that the TADS parser runs on the server and was largely bypassed in the current implementation.

Secondly, it seems that the interactive drama poetics could use some further thought, particularly around the notion of player agency. It was encouraging to see that the two different stories used in this study led to significantly clearer story-level goals. However, the clarity of these story-level intentions, as reported by users, did not correlate positively or negatively with world-level agency, as the poetics suggests that it should. Even more importantly, Skald significantly improved the ease with which users could recognize the affordances of the game world and construct commands. Yet this improvement in the "material cause" available to their choices did not have a significant impact on their feelings of world-level agency. It seems that some other variable having to do with the nature of the players themselves may be at work here.

APPENDIX A: PARTICIPANT RECRUITING EMAIL

Subject: Research participants needed: Play a new game today!

This is a request for volunteers to participate in a fun Masters research study.

THE RESEARCH:
As part of my Computer Science degree work at the University of Hawaii, I have developed a new menu-driven user interface for text-based adventure games. This is a classic computer game genre dating back to the late 1970s. Would a different user interface make these old games more accessible to modern audiences, or would it destroy the essence of the genre?

I need some live players to help me find out!

THE GAMES:
* "Captain Fate"
How is a superhero going to save the city from peril if he can't first find a place to change his clothes?

* "The Queen's Heart"
Through the long years, a lone goblin knight tends to his slumbering queen... until a little human girl falls down a well and into his world.

THE STUDY:
Anyone can play! You can participate online from any computer.

You'll answer a short survey about your gaming and computing background, and then play though the two text-based games, answering a short survey after each game session. Each of the text-based games will have a different user interface. The full study should take less than 60 minutes to complete.

To get started, click this link:
http://demo.zach.tomaszewski.name

The study will only run until 20 Dec, so act now!

WHY SHOULD I CARE?
There are so many good reasons for you to participate:
* HELP a hungry grad student finally graduate.
* EXPERIENCE something new: Two new games!
* DISCOVER something old: Text-based adventure games and

interactive fiction have a 30+ years history and are still being produced by a small but active community. Have you ever played one?
* SUPPORT local research in Hawai'i: What goes around, comes around.
* RELAX and play a game, because you could probably use a break right now anyway.

Yes, you can have all of this and MORE! Click the link above.

QUESTIONS?
If you have any questions or want to report any problems, you can contact me at: ztomasze@hawaii.edu

Feel free to forward this email to others you think might be interested. Thank you for your time!

--Zach Tomaszewski
ICS MS student

APPENDIX B: INFORMED CONSENT FORM

You are about to participate in a research study regarding user interfaces for text-based computer games. If you choose to continue, you will be asked to:

1. Complete a short background survey.
2. Play a text-based game named *Captain Fate*
3. Complete a 1-page response survey about your experience of the game
4. Play a text-based game named *Queen's Heart* that uses a different user interface
5. Complete the same response survey again, plus a few comparison questions.

The complete study should take you about 60 to 90 minutes to complete.

Your participation is anonymous, and you are free to quit at any time.

Details:

RESEARCH:
This research project is being conducted as a component of a master's thesis. The purpose of the project is to investigate the effects of different user interfaces (including input controls and output presentation style) on the experience of playing a text-based interactive fiction game.

PARTICIPATION:
Your participation should take approximately 70 minutes. Your participation will consist of:

- filling out a form on background information about yourself (10 minutes)
- playing a short, text-based computer game (20 minutes)
- filling out a form regarding your experience of the game (5 minutes)
- playing a different text-based computer game (with a different user interface) (25 minutes)
- filling out a form regarding your second experience of the game (5 minutes)
- filling out a form comparing the two experiences (5 minutes)

Participation in this research project is completely voluntary. You are free to withdraw your participation at any time during the duration of the project with no penalty or loss of benefit to which you would otherwise be entitled.
RISKS:

The investigator believes that, beyond the risks associated with normal computer use, there is little or no risk to participating in this research project.

BENEFITS:
Participating in this research may be of no direct benefit to you. It is believed, however, that the results from this project will help improve the design of interactive fiction systems.

CONFIDENTIALITY:
Your name will not be collected. You may optionally provide an email address to learn about the final results of the study. Your participation in the project will be confidential to the extent allowed by law. Agencies with research oversight, such as the UH Committee on Human Studies, have the authority to review research records. Any identifying record of your participation will be destroyed at the end of the project.

QUESTIONS:
If you have any questions regarding this research project, you can contact the investigator:

Zach Tomaszewski
ztomasze@hawaii.edu / (808) 923-5372.
http://zach.tomaszewski.name/argax/

If you have any questions regarding your rights as a research participant, you can contact the UH Human Studies Program:

uhirb@hawaii.edu
http://hawaii.edu/irb/

Please print this form for your records.

I have read and understand the above information, and I agree to participate in this research project.

[ ] Yes

How did you hear about this study?

( ) An email list. Please specify: _____

( ) A personal email.

( ) Other. Please specify: _____

Is this your first time participating in this study or playing the two games?

( ) Yes, this is my first time.

( ) No, I have played one of the games or participated in this study in the past.

APPENDIX C: BACKGROUND SURVEY

1. On average, I spend the following number of hours each day playing computer or online games on a personal computer (PC).

( ) Six or more hours a day.
( ) Three to five hours a day.
( ) One to two hours a day.
( ) Less than one hour a day
( ) I rarely or never play computer games.

2. On average, I spend the following number of hours each day playing digital games on platforms other than a personal computer—such as on gaming consoles, mobile phones, hand-held devices, arcade games, etc.

( ) Six or more hours a day.
( ) Three to five hours a day.
( ) One to two hours a day.
( ) Less than one hour a day
( ) I never or rarely play digital games on non-PC platforms.

3. I have played approximately the following number of different games using a personal computer:

( ) Over 100.
( ) 50 to 100.
( ) 20 to 50.
( ) 5 to 20.
( ) 1 to 5.
( ) I have never played a game on a personal computer.

4. I have played approximately the following number of digital games on platforms other than a personal computer (such as on a gaming console, mobile phone, hand-held device, arcade game, etc.):

( ) Over 100.
( ) 50 to 100.
( ) 20 to 50.
( ) 5 to 20.
( ) 1 to 5.
( ) I have never played a digital game on a non-PC platform.

5. How often do you use a command line interface? (Examples of command line interfaces include the Windows Command Prompt, MacOS's Terminal, SSH, MUDs, etc.)

( ) I use a command line at least once a week for many different tasks.
( ) I use a command line at least once a week for a small number of tasks.
( ) I use a command line a few times a year, or I have regularly used a command line in the past.
( ) I rarely use a command line.
( ) I have never used a command line (or, I do not know what a command line interface is).

6. Please indicate how familiar you are with each of the following computer game genres according to the following scale:

5 - I have played a number of games in this genre
4 - I have played at least one game in this genre for many hours
3 - I have briefly played or watched someone else play a game like this
2 - I know what this is, but I have never played one
1 - I do not know what this is

a. Text adventure / interactive fiction 5 4 3 2 1
(*Adventure, Zork,* games by Infocom or Legend, independent IF, etc.)

b. Graphical adventure games 5 4 3 2 1
(*Sierra* games such as *King's Quest* or *Space Quest; Monkey Island* series, etc.*)*

c. Computer roleplaying games 5 4 3 2 1
(*Diablo, Grand Theft Auto, Skyrim, Final Fantasy*, etc)

d. MUDs and/or MOOs 5 4 3 2 1

e. MMORPGs 5 4 3 2 1
(*World of Warcraft, EVE Onine, Everquest,* etc.)

f. Social simulations 5 4 3 2 1
(*The Sims*, etc.)

g. First-person shooters 5 4 3 2 1
(*Doom, Quake, Half-Life*; *Tomb Raider*, etc)

7. Your gender: Male / Female

8. Your age category:

18 – 24 25 – 34 35 – 44 45 – 54 55 – 64 65 – 74
75+

9. Highest level of education completed:

( ) Doctoral / Professional (PhD, MD, etc)
( ) Graduate School (Masters, etc.)
( ) University / College (Bachelors, etc.)
( ) Community College (Associates, etc)
( ) High school
( ) None of the above

APPENDIX D: RESPONSE SURVEY

Please answer the following questions about the game session that you just finished playing:

1. It was clear which objects I could interact with in the game world.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

2. I knew which actions I could perform in the game.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

3. I was able to able to construct commands that the game understood.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

4. I was sufficiently able to direct my character's actions in the game world—such as move from place to place, manipulate objects, affect other characters, etc.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

5. I usually knew what I was expected to do in the game (even if sometimes I had to figure out exactly how to accomplish it).

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

6. The game session had a story-like structure.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

7. The game contained puzzles or situations that required some thought or exploration to overcome.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

8. I believe that the game probably would have had a different outcome had I performed different actions.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

9. I enjoyed playing this game.

5 - strongly agree 4 - agree 3 - neutral 2 - disagree 1 - strongly disagree

10. What was the LEAST enjoyable aspect of this game?

11. What was the MOST enjoyable aspect of this game?

12. Comments?

If you have any other comments to share about the game or this survey that didn't seem to fit elsewhere, please enter them here:

APPENDIX E: COMPARISON SURVEY

You used a different user interface for each game. In the "command line interface", you typed in commands expressing what you wanted to do. In the "menu-based interface", you clicked on links in the text and selected the command from a menu.

Which interface did you find easier to use given the following criteria?

1. Which interface did you find easier to use?

( )	( )	( )	( )	( )
command line (definitely)	command line (by a small margin)	**no preference** (about the same)	menu-based (by a small margin)	**menu-based** (definitely)

2. Which interface did you find faster to use?

( )	( )	( )	( )	( )
command line (definitely)	command line (by a small margin)	**no preference** (about the same)	menu-based (by a small margin)	**menu-based** (definitely)

3. Which interface was more enjoyable to use?

( )	( )	( )	( )	( )
command line (definitely)	command line (by a small margin)	**no preference** (about the same)	menu-based (by a small margin)	**menu-based** (definitely)

4. Which interface made the game more interesting or engaging?

( )	( )	( )	( )	( )
command line (definitely)	command line (by a small margin)	**no preference** (about the same)	menu-based (by a small margin)	**menu-based** (definitely)

5. If you were to play a third text-based game, which interface would you prefer to use?

( )	( )	( )	( )	( )
command line (definitely)	command line (by a small margin)	**no preference** (about the same)	menu-based (by a small margin)	**menu-based** (definitely)

6. Comments? Any additional comments regarding your preference for one interface over another:

7. (Optional) If you would like to receive a single email summarizing the final results of this study, enter your email:

Thank You!

APPENDIX F: RESPONSE SURVEY COMMENTS

From the Response Survey (Appendix D), the following are the collected responses to the question "Comments? If you have any other comments to share about the game or this survey that didn't seem to fit elsewhere, please enter them here."

Group TS - *Captain Fate*, Text UI

- sorry if my review was skewed by the slowness with which it ran. it's probably a problem on my end. but then again, i'm not the only person with such problems :)
- Make more intuitive commands. Make secondary actions possible like interacting with other customers when they mock you.
- I found this form of game interesting, because I actually formed text-based game in my ICS 313 (Programming Language Theory) class with a group last semester in LISP, so I was glad to see something like this again. I was expecting a bit more tries, since I lost and I didn't know what the entire goal of the game was, but if it was to change clothes in public, I wish the game was a little bit more elaborate than that. Since the game is almost linear, there should also be more directions in the game to traverse.
- Games like this probably could be more easily designed by simply recording what some users attempt to do as they are presented with the situations--players want the game to feel like it has intuitive controls (or in this case commands). Having to consult the help menu for the specific grammar of each command pretty much made immersion in the game world impossible.
- The HELP command, while useful, could use more information. I was trying to figure out what commands I could use and on more than one occasion, thought I hit a dead end.
- I think I just don't enjoy text-based games.

Group ST - *Captain Fate*, Skald UI

- When I was wearing my suit (I also dropped my ordinary clothes), I examined myself and it said I looked "the same as usual" which I found strange. Also I wonder how the scoring worked. When the game ended I had earned 2 out of 2 points, but i didn't know we were earning points.
- WHY COULDNT I PAY BENNY??????
- the action commands were confusing, especially trying to figure out how to go into the restroom once I got the key, also, is the lack of money to pay for coffee intentional. Why can't I attack Benny. Interesting game though, it was fun
- The ending was a little abrupt, and I wonder about some of the other things I could have tried, but that is inevitable with my time constraints.

- For a game where you can't see the world, it would be nice to have some more of the little things in there.

Group TS - *The Queen's Heart*, Skald UI

- this game ran ten times faster than the other one.
- I'd like to play this one again to see if I can save the queen.
- Holy shit, that was an excellent story.
- The UI definitely was better, but the game was still basically unplayable. I would never inflict it on an undergraduate; I have a pretty high patience with games. I've played a few MUDs but never for very long, but never for very long. These sorts of games place a very high value on very specific physical spaces... north north south south, when text-based games just are not geared toward strict spatial limits (obviously). They should be driven by purpose-driven spaces. I kept wanting to quick travel, or just press E or W to move east and west (rather than typing east or west). This may seem like a minor complaint (especially given the shortness of the game, or at least my experience with it) but since the descriptions of the hallway after the mole disappears (I failed to eat it :/) were so scarce it felt pointless to manually type out the words or click twice to get to the East/West option. In some I preferred the single buttons at the top for movement between zones from the first game, even though this game certainly had a lot more going on and a better UI aside from that. I still feel like the "glammer" (or however it was spelled) mechanic needed to be introduced earlier. I spent about 3 minutes trying to get a sense of how to talk with the girl (who falls through the hole) before just giving up. I had no idea why she couldn't see me and why I wasn't supposed to directly interact with her.
- I really wish that I could replay this game.
- I kind of want to play it over again so I can see if there's a way to rescue the girl instead of (totally accidentally) killing her.

Group ST - *The Queen's Heart*, Text UI

- The story ended very abruptly causing me to see the game as more of an interactive story, rather than an adventure game. I was very immersed in the story.
- I really like the game. The ending was unique and sad, but I liked it.
- not sure if I succeeded or failed in this game
- I was not too clear as to what the goal of the game was. Some more flavor text in the beginning could go a long way. Text based games are all about the writing in my opinion.

APPENDIX G: COMPARISON SURVEY COMMENTS

From the Comparison Survey (Appendix E), the following are the collected responses to the question "Comments? Any additional comments regarding your preference for one interface over another:"

Group TS

- the menu-based game had a much denser story and ran much, much faster in my browser. these factors definitely influenced my decision -- honestly, i work in the command line 40 hours a week and i *wanted* to like the first one better, but it was too slow :(
- The command line was more challenging and more fun. The only thing I liked about the menu better was that I knew what I was able to do better
- I'd like to combine command line with an action window so what I can do is viewable, but I can still type what I want to do.
- I definitely liked the command line interface much better, since I have a game in LISP in one of my previous projects. I like the use of the keyboard in playing the game, since it involves more interaction with the audience. To compare these two interfaces to regular lectures and inverted classroom, the command line interface represents the inverted classroom, which invokes more involvement with the audience, whereas menu-based interface represents regular lectures, where there is basically no interaction with the audience (based on a set of pre-set choices).
- Menu is simple and makes things clearer, useful in a more linear game. With the command line, the players have more flexibility to experiment, so it seems better for exploration/puzzle games.
- The menu-based UI is definitely better (except for movement from zone to zone). However, menus alone can not save either game since they seem to be suffering from poor game design (imho obviously) and very questionable decisions as far as introducing key game mechanics.
- It's a lot easier to figure out what to do when the possible options are put out before you, even if it does feel more realistic not to have them there.

Group ST

- When you can see all the commands in a bar like in Captain Fate, it gets really overwhelming when choosing what you should do. Although I think that the type of interface tends to change the feel of the game. In Captain Fate I felt like there was no object, I just chose things randomly from the menu, things that I wanted

to do. In The Queens Heart I always felt that there was some specific task that I was trying to accomplish, so I formed specific commands that helped me achieve what I thought I was supposed to do.

- The menu interface was really complicated to understand. I didn't know if i should click in the words or figure out which menu item i should click next
- I like the first game better, there was a clear task.
- The command line style was more compact and easier to use overall.
- The goblin game was more interesting and engaging, mysterious, but also less satisfying and conclusive. I am not sure if that is because of the CLI, or because of the content.

WORKS CITED

Aristotle. *Poetics*. Trans. S. H. Butcher. Ed. Francis Fergusson. New York: Hill and Wang, 1961.

Bolter, J. David and Michael Joyce. "Hypertext and Creative Writing". *Proceedings of ACM Hypertext*. Chapel Hill, NC, 1987. pp 41-50

"Brass Lantern: The Adventure Game Website" <http://www.brasslantern.org/> Last accessed: 26 Jan 2015.

Cadre, Adam. *Photopia*. 1998. <http://adamcadre.ac/if.html> Last accessed: 26 Jan 2015.

Cooper, Alan, Robert Reimann, and David Cronin. *About Face 3: The Essentials of Interaction Design.* Indianapolis, IN: Wiley Publishing Inc., 2007.

Eve, Eric. *Captain Fate*. TADS source code available from <http://www.ifarchive.org/indexes/if-archiveXgamesXsourceXtads.html> 07 Sep 2005.

Firth, Roger, and Sonja Kesserich. *Inform Beginner's Guide*. Aug 2004. <http://inform-fiction.org/manual/about_ibg.html> Last updated: 17 Apr 2013.

Gibson, James J. "The Theory of Affordances." *Perceiving, Acting, and Knowing: Toward an Ecological Psychology.* Eds. Robert Shaw and John Bransford. Hillsdale, NJ: Lawrence Erlbaum Associates, 1977.

"Home: Inform" <http://inform7.com/> Last accessed: 27 Jan 2015.

"Interactive Fiction Archive" <http://ifarchive.org> Last accessed: 26 Jan 2015.

"Interactive Fiction Competition" <http://ifcomp.org/> Last accessed: 26 Jan 2015.

"Interactive Fiction Database" <http://ifdb.tads.org/> Last accessed: 26 Jan 2015.

Jota (Admiral) and Grunk. *Lost Pig.* 2007. <http://grunk.org/lostpig/> Last accessed: 26 Jan 2015.

Klimas, Chris. "Twine: A Tool for Creating Interactive Stories" <http://twinery.org> Last accessed: 26 Jan 2015.

Mateas, Michael. "A Preliminary Poetics for Interactive Drama and Games." *First Person: New Media as Story, Performance, and Game.* Eds. Noah Wardrip-Fruin and Pat Harrigan. Cambridge, MA: MIT Press, 2004. Reprinted at: <http://www.electronicbookreview.com/thread/firstperson/mateas>

Mehta, Manish, Andrea Corradini, Santiago Ontañón, and Peter Juel Henrichsen. "Textual vs. Graphical Interaction in an Interactive Fiction Game." *ICIDS'10 Proceedings of the Third joint conference on Interactive digital storytelling*, 2010. pp 228-231. <http://sites.google.com/site/santiagoontanonvillar/publications/pdfs/icids2010_short_v14.pdf>

Montfort, Nick. *Twisty Little Passages: An Approach to Interactive Fiction.* Cambridge, MA: MIT Press, 2005.

Nelson, Mark J, Michael Mateas, David L. Roberts, and Charles L. Isbell Jr. "Declarative optimization-based drama management in interactive fiction." *IEEE Computer Graphics and Applications*, 26(3): 32-41, 2006. <http://www.kmjn.org/publications/DODM_IEEECGA-abstract.html>

Nelson, Mark J, and Michael Mateas. "Search-Based Drama Management in the Interactive Fiction *Anchorhead*." *Proceedings of the First Artificial Intelligence and Interactive Digital Entertainment Conference*, 2005. <http://www.kmjn.org/publications/Anchorhead_AIIDE05-abstract.html>

Norman, Donald A. *The Design of Everyday Things.* New York: Basic Books, 2002.

Plotkin, Andrew. "Quixe: a Glulx VM interpreter written in Javascript." <http://eblong.com/zarf/glulx/quixe/> Last updated: 27 Mar 2014.

"Parchment." <https://code.google.com/p/parchment/> Last accessed: 27 Jan 2015.

"Playing Interactive Fiction: Inform" <http://inform7.com/if/interpreters/> Last accessed: 27 Jan 2015.

"Project Aon." <http://www.projectaon.org/en/Main/Home> Last updated: 25 Jun 2014.

Sali, Serdar, Noah Wardip-Fruin, Steven P. Dow, Michael Mateas, Sri Kurniawan, Aaron A. Reed, and Ronald Liu. "Playing with Words: From Intuition to Evaluation of Game Dialogue Interfaces" *Proceedings of the Fifth International Conference on the Foundations of Digital Games*. ACM, 2010. <http://www.cs.cmu.edu/~spdow/files/Agency-FDG10.pdf>

Sharma, Manu, Santi Ontañón, Manish Mehta, and Ashwin Ram. "Drama Management and Player Modeling for Interactive Fiction Games." *Computational Intelligence,* 26(2):183-211, 2010. <http://www.cc.gatech.edu/faculty/ashwin/papers/er-09-10.pdf>

Short, Emily. *Galatea*. 2000. <https://emshort.wordpress.com/my-work/> Last accessed: 26 Jan 2015.

"TADS - the Text Adventure Development System, an Interactive Fiction authoring tool." <http://tads.org/> Last accessed: 27 Jan 2015.

"Teach with Inform: Inform" <http://inform7.com/teach/teach/> Last accessed: 27 Jan 2015.

Tomaszewski, Zach, and Kim Binsted. "A Reconstructed Neo-Aristotelian Theory of Interactive Drama." *Computational Aesthetics: Artificial Intelligence Approaches to Beauty and Happiness: Papers from the 2006 AAAI Workshop.* Menlo Park, CA: AAAI Press, 2006. pp. 103-106. <http://zach.tomaszewski.name/argax/pubs/2006-TomaszewskiBinsted-Drama.pdf>

Tomaszewski, Zach. "Game: The Queen's Heart." Global Game Jam 2013. <http://2013.globalgamejam.org/2013/queens-heart> Last updated: 28 Jan 2013.

Tomaszewski, Zach. "IxD: Overview to the Interaction Design Process for the Skald UI." <https://code.google.com/p/skald/wiki/IxD> Last updated: 30 Aug 2012.

Tomaszewski, Zach. "skald - A different approach to interactive fiction." Google Project Hosting. <https://code.google.com/p/skald/> Last updated: 30 Aug 2012.

Tomaszewski, Zach. *Marlinspike: An Interactive Drama System*. Dissertation. University of Hawaii-Manoa, 2011.

Weyhrauch, Peter. *Guiding Interactive Drama*. Dissertation. Carnegie Mellon University, 1997.

Williams, Roberta, and Ken Williams. *Mystery House*. On-Line Systems, 1980.

Winograd, Terry. "SHRDLU". <http://hci.stanford.edu/~winograd/shrdlu/>
Last accessed: 26 Jan 2015.