

Transparency or Opacity? Re-thinking Video Game Interfaces

Etienne Brunelle-Leclerc

A Thesis
in
The Department
of
Design and Computation Arts

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Design at
Concordia University Montreal, Quebec, Canada

October 2018

© Brunelle-Leclerc, 2018

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Etienne Brunelle-Leclerc

Entitled: Transparency or Opacity? Re-thinking Video Game Interfaces

and submitted in partial fulfillment of the requirements for the degree of

Master of Design

complies with the regulations of the University and meets the accepted standards with respect to originality and quality.

Signed by the final Examining Committee:

Jonathan Lessard Chair

Rilla Khaled Examiner

Pippin Barr Examiner

Jonathan Lessard Supervisor

Approved by _____
Chair of Department or Graduate Program Director

_____ 2018

Dean of Faculty

ABSTRACT

Transparency or Opacity? Re-thinking Video Game Interfaces

Etienne Brunelle-Leclerc

This thesis presents the results of a two year research-creation project that sought to address a gap in the theorization of video game interfaces. The conceptual framework of game interface design borrows several concepts from more established schools of user experience design. However in doing so, it also imports the biases that are built into these concepts, i.e. that interfaces should always maximize transparency and control. While this holds true for regular software, it does not apply to game interfaces where lapses in transparency and control can be repurposed as sources of challenge. This leaves designers with no way to adequately represent the interface's contribution to gameplay. This project used a research-creation approach to investigate the merits of possibilities that run against the grain of standard interface design practice. The creative part of the research (conducted in collaboration with the Lablablab team) has produced *Hammurabi*, a game that leverages the inefficiencies of its interface as the centerpiece of its gameplay. Building upon this success, the reflexive part of the research offers a new theoretical perspective that proposes to frame game interfaces in terms of opacity as opposed to transparency and control as well as concepts and design strategies that will assist the work of future designers.

ACKNOWLEDGEMENT AND DEDICATIONS

À mes parents,

TABLE OF CONTENT

CHAPTER 1: APORIAS IN CURRENT GAME INTERFACE DESIGN THEORIZATION	1
1.1 INTRODUCTION	1
1.2 GENERAL APPROACHES TO GAME INTERFACES	6
1.3 INFORMATION FLOW IN GAME INTERFACES	10
1.4 DESIGN FRAMEWORKS AND DESIGN RESEARCH	11
1.4.1 The Usefulness of Frameworks	12
1.4.2 Design Frameworks as Research Results	13
CHAPTER 2: MOVING BEYOND STANDARD INTERFACE DESIGN	15
2.1 INTRODUCING HAMMURABI	15
2.2 CONCEPTUALIZATION : REMAKING HAMURABI	16
2.2.1 Viziers	19
2.2.2 Ministries	19
2.2.3 Approval	20
2.2.4 Crises	21
2.3 PROTOTYPING & ROLEPLAYING	22
2.4 IMPLEMENTATION	27
2.4.1 Hammurabi's Subjective Interface	28
2.4.1.1 Procedural Dialog Generation with Expressionist	28
2.4.1.2 From Generativity to Modularity	30
2.4.1.3 CFG Templates	31
2.4.2 Hammurabi's Graphical User Interface	34
2.4.2.1 Resource Gauges	34
2.4.2.2 Resource Sliders	38
2.4.3 The Contrarian Interface	42
2.5 RECEPTION	43
CHAPTER 3: TOWARDS AN EXPANDED UNDERSTANDING OF GAME INTERFACES	45
3.1 BEATING HAMMURABI	45
3.2 DESIGNING FOR OPACITY	46
3.3 SUMMARY & DIAGRAM	61
CHAPTER 4: CONCLUSION	62
BIBLIOGRAPHY	65
WORKS CITED	68

CHAPTER 1: APORIAS IN CURRENT GAME INTERFACE DESIGN THEORIZATION

1.1 INTRODUCTION

In digital media, the user interface is the particular piece of technology (either software, hardware or a combination of both) that allows the user to communicate with the computer. While the design of the interface is critical to any computer application, it is especially sensitive in video games because they vary so widely in both purpose and structure compared to regular software. However, interface design is an area where the gaps in our current understanding of video game design are especially apparent. This might seem counterintuitive given the wealth of information and design knowhow available in related fields such as human-computer interaction and user experience design. As we shall see, some of these concepts like Mihály Csíkszentmihályi's *flow* and Jesse James's Garrett's user-centered design strategies for computer applications have indeed been explored in the video game design literature, but these efforts have yet to acknowledge the significance of the radically different priorities that separate the design of video game interfaces from that of regular software.

The inadequacy of existing game design theory is not limited to interfaces. Video games cut across disciplines and domains of knowledge in ways that makes them unnervingly resistant to theorization. From the perspective of a researcher studying the play and design of video games, it often seems as though they take many forms and serve many purposes, all at once. This muddled status has led the game design community to borrow concepts from several disciplines in order to constitute its own theoretical framework. While this approach has enriched our understanding of how video games work in many deeply insightful way, it is not without its problems. This initial push for theorization, which ran parallel to the professionalization of video

game design and was thus spurred by an urgent need to accumulate a critical vocabulary, has left us with a vast array of concepts to draw from when giving form to our inquiries. Still, we should not lose sight of the fact these concepts originate from other fields of knowledge each with their own history, conventions and epistemological agendas. Therefore, their applicability to video game design is limited. In order to drive video game design theory forward, designers and researchers need to look closely at how the concepts that they use to frame their inquiries apply to games specifically. Rather than to study games from the outside in, using concepts in a top down manner to guide our perception of what constitutes a video game, we ought to do the reverse and examine the way games reflect upon the concepts we use to describe them in order to arrive at a more fine-grained picture of how individual components impact the player's experience.

Several researchers have taken a similar approach to the study of game interfaces and interactions. Nicole Lazzaro's research on emotion in gameplay (2004) does attempt to address the interface's contribution to the aesthetics of gameplay, but only conceives of it in terms of enjoyment versus frustration which still frames the game interface as merely a facilitator of interaction. Steve Swink's work on 'game feel', which he defines as "the tactile, kinesthetic sense of manipulating a virtual object", offers a set of concepts and methods to design how a game's controls should feel to the player (2008, p.9). In defining and exploring this "aesthetic of control", Swink establishes a bridge between one of the main functions of the interface—control—to the aesthetic experience of the game. In her 2013 book titled *Gameworld Interfaces* (MIT Press), Kristine Jørgensen argues that a game's virtual world is a part of its interface since it is an "informational environment designed to support gameplay" (2013, p.9), indicating that the relationship between the functionality of the game interface and the aesthetic experience of the player is very much a multilayered and complex one.

These individual efforts show that the video game design community is steadily moving in a more holistic direction. That being said, the extent to which this work has trickled down to the manner in which game design is being taught and practiced is unclear. In my own experience as an undergraduate student in the field of game design, the textbooks that were referenced subscribed to a predominantly functional conceptualization of game interfaces. This is problematic not because it is incorrect—after all, games interfaces must function in ways that are similar to that of regular software—but rather because it obscures the aesthetic dimension of interface design in video games. As we shall see, many of the field’s foundational textbooks display a surface understanding of the interface’s contribution to the player’s experience. Even entire manuals solely dedicated to the design of video game interfaces regard it as a purely functional construct and leave much of its expressive possibilities untapped, save for industry-tested recipes. Moreover, the theories advanced by Swink and Jørgensen have well defined domains of applicability: Swink states that his definition of game feel only applies to game that prominently feature “real-time control” within a spatial simulation (2008, p.8). Jørgensen’s case studies cover a broad range of games, but her definition of the gameworld interface as a composite of representational and informational elements (2013, pp.143-144) is most relevant to games that are played from a first person or a third person point of view, as is reflected in the composition of her corpus. This is not meant a criticism of Swink and Jørgensen’s work. Video games come in many shades and forms and we should be equally interested in what sets them apart as we are in what connects them. That being said, the limitations of their respective theories make them less useful in understanding the relationship between the design of the interface and the player’s experience in games where the spatial representation is more abstract and where the gameplay does not revolve around the moment-to-moment control of an avatar. This research aims to address this issue by providing a more high-level understanding of how the interface affects gameplay as well as a set of concepts that will help future designers fine-tune the interface of their games to achieve the desired effect. I don’t propose to overhaul

the current formalization of interfaces design within the game design literature. Rather, I hope that the framework that I propose in this thesis can help bridge the gap between the aesthetic dimension of game design and the functional description of game interfaces.

However, this cannot be accomplished simply by analyzing existing games as they rarely deviate from standard interface design practice. In order to see beyond this horizon, we need to *make* games that actively challenge the assumptions that are built into our existing frameworks. This is where research creation shines as a method. According to Chapman & Sawchuck, “each and every research-creation project [...] carries the possibility of acting as an intervention in its own right in terms of the specific fields of inquiry, practice, history, et cetera in which it is embedded.” (2012, p.19). This kind of “epistemological intervention” may be what game interface design needs to escape its current state of theoretical entanglement, so long as it is followed up by a reflexive theorization effort that attempts to bridge the gaps between theory and practice. In practice, this can take the form of an exercise in contrarian design, similar in spirit to Katerina Kamprani’s ‘Uncomfortable’ series (2017) albeit with a sharper focus. Rather than explore how everyday objects can be made to evoke discomfort, the goal is here to investigate the merits of so-called ‘bad ideas’ by breaking away from the courses of actions that are prescribed by design theory. If the resulting designs prove to be effective nonetheless, then they point to a flaw in the theory which researchers can then attempt to correct. This approach resembles critical design (Dunne, 2006) abusive design (Sicart & Wilson, 2010) and adversarial design (DiSalvo, 2012) in that it “challenges the conventions of normative game design” (Sicart & Wilson, 2010). However, it does not take an outwardly militant stance regarding the ideological dimension of design. The purpose of this research is merely to help game designers design games.

This thesis will be divided in four chapters. In the rest of Chapter 1, I will review the conversation around interface design within the video game design literature in order to expose the limitations of prevailing theories, namely their reliance on concepts borrowed from other schools of user experience design that leave us with an incomplete picture of the interface's contribution to the player's experience. I will then examine how this contribution can be framed in terms of information flow. Next, I will densify the epistemological and methodological foundations of this research by looking at other research projects that sought to address weaknesses in existing design frameworks. Chapter 2 will feature an in-depth examination of the creative part of this research. I will explain how my thesis game *Hammurabi* (made with the Lablablab research group) challenges established interface design practice and recount the challenges that the design team faced while operating outside of these conventions. In doing so, I will draw upon interviews conducted with fellow designers and developers Christopher Tan and Marc-Antoine Jetté-Léger in which we reflected on the game's pre-production and reception. *Hammurabi*'s development was archived in the form of a git repository which I was able to access after the fact in order to reconstruct the iterative process that led to the final game¹. After an extensive overview of the game's eighteen month development cycle, I will conclude with an assessment of the game's reception. Last but not least, Chapter 3 will feature this research's main theoretical contribution as I attempt to package the lessons of *Hammurabi*'s development into a game interface design framework that I hope will empower designers that are interested in working outside of established frameworks. But before we can come to that, let us first review the state of discourse in regards to video games and interfaces.

¹ This approach takes after the one that Khaled *et al.* outlined in their 2018 on applied game design research methodology.

1.2 GENERAL APPROACHES TO GAME INTERFACES

This section will serve as a review of how the notion of interface is currently construed within the video game design literature, especially at an introductory or undergraduate level. We will look at several game design textbooks as well as more scholarly perspectives in the works of game researchers Kristine Jørgensen, Jesper Juul, Jonathan Lessard and Dominic Arsenault in order to accumulate an understanding of the concepts that are used to frame the design of video game interfaces. This information will later serve as the basis for a critique of existing video game interface frameworks.

Ernest Adams's *Fundamentals of Game Design* (New Riders, 2010) is one of the founding volumes of video game design. In the chapter dedicated to the user experience, Adams acknowledges the importance of the interface's contribution to the player's experience, stressing its "enormous effect on whether the player perceives the game as satisfying or disappointing, elegant or graceless, fun or frustrating" (Adams, 2010). However, Adams' actual interface design framework is derivative, building on "interaction models" and "camera models" prevalent in existing games with a sharp focus on navigation which was, by the author's admission, a feature nearly ubiquitous in video games at the time of the book's writing. Furthermore, Adams frames the design of the video game interface with concepts borrowed from another, more established school of experience design, citing information architect Jesse James Garrett's book *Elements of User Experience* (New Riders, 2002). That being said, Adams also calls attention to the fact that "a game's user interface plays a more complex role than does the UI of most other kinds of programs" (Adams, 2010). Similarly, he remarks that interfaces meant for tools and ones meant for entertainment will inevitably have different design priorities, namely that "a video game interface doesn't tell the player everything that's happening inside the game, nor does it give the player maximum control over the game" (p. 257). Yet, Adams does not push

these ideas further, even admonishing “unnecessary innovation” in interface design on the grounds that video games genres have evolved a “practical set of feedback elements and control mechanisms suited to their gameplay” (p. 257). The approach developed in the coming chapters will expand on Adams’s insight into the specificity of the video game interface while eschewing its reliance on established gameplay models so as to tighten the focus around *what* video game interfaces can do as opposed to *how* they usually do it.

Another seminal video game design textbook is *The Art of Game Design: A Book of Lenses* by veteran game designer Jesse Schell (2015). *The Art of Game Design* takes its subtitle from the 113 design lenses that Schell introduces throughout the book. Each lens consists of a set of questions meant to emphasize how one’s design relates to a central idea or design principle. In the chapter titled “Players play games through an *Interface*” (original emphasis), Schell invokes, among others, the lenses of transparency and control. On the topic of transparency, Schell states that “Players project themselves into games and on some level disregard that the interface is there at all, unless it suddenly becomes confusing” (p.259). Through the lens of transparency, an interface that ostensibly makes its presence felt to the player is framed as immersion-breaking and thus a failure in design. This view rests on a shallow understanding of the player’s experience where ‘immersion’ is likened to a kind of flow state when Laura Ermi & Franz Mäyrä (2005) as well as Dominic Arsenault (2005) have shown that immersion is multi-faceted and doesn't only emerge from "seamless interaction". The video game design literature makes abundant reference to flow states, which describe an optimal state of consciousness characterized by “a sense that one is engaging challenges at a level appropriate to one’s capacities” as well as “clear proximal goals and immediate feedback” (Nakamura & Csikszentmihalyi, 2002). Jesse Schell explicitly references the concept of the flow channel as defined by Csikszentmihalyi in his discussion of focus (2015, p.119). One of the design lenses associated with this discussion is the *Lens of Flow*. In *Fundamentals of Game Design*, Ernest

Adams claims that a well-balanced game should consider the player's skill level so as to produce an experience that is neither too easy or too hard (Adams 2010, p.359).

My thesis game *Hammurabi* demonstrates that the player's experience doesn't have to be optimal to be enjoyable and that game interfaces can enhance gameplay in manners that goes beyond functionality (this topic will be discussed at length in chapters 2 and 3). In his discussion of control, Schell asserts that "the goal of the interface is to make the player feel in control of the experience." Here, the underlying assumption seems to be that the more agency players have over the outcome of a game, the more they will care about it. This is rather misleading, since giving the player total control over the outcome would obviously result in a game where failure states are non-existent. In the words of Jesper Juul, most games involve "the player working toward a goal, either communicated by the game or invented by the player, and the player **failing** to attain that goal" (2013, p.14, added emphasis). Although a minority of games manage to find an audience without explicitly challenging the player (walking simulators come to mind), the fact is that the overwhelming majority of games do, which shows how fundamental the relationship that unites challenge, failure and reward is to the appeal of video games as a medium. This casts the notion of player agency in a new light as it rather seems as though players prefer to *earn* their ability to control the game's outcome rather than have it be surrendered to them from the beginning. What's more, a game like *The Sims* (Maxis, 2000) leverages the transparency of its interface and the level of control that it provides in a manner very different from *The Walking Dead* (Telltale Games, 2012). Transparency and control are thus better understood as key variables of the player's experience that should be tuned with care and precision in order to produce the desired effect, rather than simply maximized at every opportunity.

Game Design Essentials: Interface Design by Jeannie Novak and Kevin Saunders (Delmar Cengage Learning, 2004) is the only game design textbook solely dedicated to the design of the video game interface. Novak and Saunders view the interface as a purely *functional* construct that is fundamentally decoupled from the game to which it is attached. Its purpose is to facilitate interaction and to allow the game to be “played as designed”. The authors also stress that an interface should be generally ‘user-friendly’ and accessible to people that suffer from disabilities, but any further discussion of the interface’s aesthetic value is limited to the visual appeal of the GUI assets. Like Schell and Adams, this way of framing interface design takes after older and more established schools of user experience design such as those found software engineering and web development where interfaces are abstracted down to their most functional constituents: control and feedback. Once again, these concepts become problematic when directly applied to video games for the simple reason that video game interfaces do much more than facilitate interaction. Rather, they modulate interaction in a manner that serves the game’s expressive aims. In the words of Ian Bogost, video games “are not tools that provide a specific and solitary end, but experiences that spark ideas and proffer sensations” (Bogost, 2008). While a video game interface still has to *function*, the set of criteria that define success and failure can vary wildly from one game to the next, which indicates that *functionality* might not be the best angle from which to approach this topic. Likewise, we cannot simply default to a criterion of efficiency since, as Ian Bogost has argued, video games can leverage inefficiencies in their interfaces to aesthetic ends (Bogost 2008 cited in Lessard & Arsenault 2016). What’s more, by thinking of games and interfaces as separated, we are effectively training ourselves to overlook the interface’s contribution to the player experience, which creates a blind spot in our thinking. As a result, we currently lack the necessary conceptual and aesthetic vocabulary to make this contribution salient.

Kristine Jørgensen's work on gameworld interface brings up another limitation of current video game interface design frameworks, namely that they subscribe to a narrow definition of what an interface is. For example, Novak and Saunders's discussion of video game interfaces only covers extradiegetic indicators such as progress bars, mini-maps, targeting reticles, menu screens, etc. However during actual gameplay the player's interactions with the game are not limited to those elements. This lack of overlap between our definitions for interface and interaction is problematic, given the former's role in mediating the latter. In order to better understand this relationship, a broader definition is needed. As Lessard and Arsenault suggested, anything that "participates in the bidirectional information flow between the player and the game" can be said to constitute an interface between the player and the game (2016). This opens a game-design oriented research field for interfaces that would look at a large spectrum of features. This project begins this work by choosing one of the most salient: information flow.

1.3 INFORMATION FLOW IN GAME INTERFACES

The authors of game design handbook *Rules of Play* (2003) propose two ways of thinking about information in games. The first refers to information theory, where information is a measure of the player's freedom of choice at a given decision point, while the second uses Celia Pearce's model in which game state information is either known to all players, known to some players, unknown to all players, or known only to the game. From a design perspective, information theory is useful in modeling the player's knowledge of the game state and extrapolating their behavior assuming rational decision-making. Since players are almost never privy to the entirety of the game state, the information theory model also allows designers to draw the line between what is known to the player and what is not. Pearce's model makes this distinction even more explicit as it allows designers to track how information is transferred across its four categories

throughout the game. Thus, in both cases the discussion of information effectively doubles as a discussion of uncertainty.

One could argue that the authors of *Rules of Play* call upon theories of information in order to get a handle on its elusive converse. As veteran game designer Greg Costikyan argued in his book titled *Uncertainty in Games* (2013), uncertainty is fundamental to the appeal of all games, be they video games, board games, card games, or even sports. Costikyan's book consists in an inventory of sources of uncertainty in games. Lessard and Arsenault identify that two of these sources are of particular relevance to the study of interfaces: hidden information, where the game deliberately provides the player with information that is incomplete or outright inaccurate, and performative uncertainty, which creates a gap between the player's intent and the result of her actions (2016). These sources of uncertainty overlap nicely with the main functions of the interface as defined by Novak Saunders: control and feedback. Indeed, the video game interfaces emerges as a powerful source of uncertainty in and of itself. Put another way, what separates a video game interface from that of a regular piece of software is that it deliberately introduces uncertainty in the flow of information between the player and the game in a manner that fosters compelling gameplay, and it is this fundamental difference that existing video game interface frameworks have failed to address.

1.4 DESIGN FRAMEWORKS AND DESIGN RESEARCH

So far, I have argued that the interface design frameworks found in game design textbooks do not account for the specificity of the video game interface. This is problematic because by framing these interfaces in a manner that does not accurately describe how they affect gameplay, we are limiting our ability to acknowledge and anticipate these effects during the

design process. However, it is one thing to point out a flaw in the current formulation of an aspect of video game design theory and another entirely to go about remedying it. In order to better understand how to address this issue, we first need stop and think about what design frameworks are, what they do and how they are produced.

1.4.1 The Usefulness of Frameworks

Ostensibly, design frameworks are sets of heuristics and concepts that assist in the design process. In their 2017 paper titled *Gaps of Uncertainty: a case for experimentation in serious game design frameworks*, Niels Quinten, Steven Malliet and Karin Coninx shed insight on exactly how they accomplish this. Referring to Donald Schön's theory of the reflexive practitioner, Quinten *et al.* frame the design process as a conversation between the designer and the materials that make up the design situation where solutions and problems emerge concurrently. In their critique of existing serious game design frameworks, Quinten *et al.* indicate a general tendency to disregard this improvisational nature of design in favor of rigid frameworks that collapse the design process into a kind of 'flowchart' where every path runs from the familiar to the predictable. Quinten *et al.* make a strong case for the importance of preserving uncertainty in the design process, noting that the process through design innovates "more fitting or novel solutions" is fundamentally exploratory, iterative and unpredictable. Consequently, the purpose of design frameworks is not to eliminate uncertainty from the design process, but rather to "encourage designers to embrace untraversed paths that can lead to unexpected yet invaluable results." Put another way, design frameworks help designers come to grips with difficult problems by providing an approximate picture of the materials that make up the design situation, but they should not overtly steer them towards a particular solution. This is especially helpful in cases where the materials are abstract, ill-defined or ontologically removed from the designer's direct experience. The authors of *Rules of Play* argue that game design is one such case when they describe it as "second order design" (p.168), meaning that designers

do not directly 'make' gameplay but rather create the mechanics that, when acted upon by players, combine to produce gameplay².

1.4.2 Design Frameworks as Research Results

Quentin *et al.* go beyond mere critique as they also propose their own serious game design framework which, in the author's perspective, encourages designers to explore the "design possibilities" that lay at the intersection of video games, learning and training. This result is especially important to the present discussion for two reasons. First, we have to acknowledge the possibility that design frameworks, no matter how open, may very well come with built-in 'biases'. Just like Quentin *et al.*'s framework is in a manner of speaking 'tailored' to the design of games that mobilize their teaching power to effect some kind of real world change, other frameworks may lead designers to formulate design problems in a manner that converges on a similarly specific set of solutions. For better or worse, resorting to a design framework will necessarily narrow the range of trajectories that a designer can take through what other researchers have referred to as the "space of possible design solutions" (Khaled *et al.* 2018). This helps better define both the scope and the interest of this research. Since design frameworks have a specific domain of applicability, the concepts that are introduced in chapter 3 of this thesis only apply to a portion of the possibility space of the video game interface. However, their value resides in how they attempt to map a part of this possibility space that current design frameworks mark as *terra prohibita*.

The work of Quentin *et al.* provides a precedent as well as a blueprint for how this kind of research can be undertaken. The authors based their serious game design framework on their own experience with designing serious games. That is to say they used design as an

² This ontological divide is further widened by the fact that video game mechanics have to be implemented in software form before the designer can assess the quality of the resulting gameplay.

experimental method to develop a way of talking about the specific type of design problems that they were interested in. This is an example of research *through* design, an aspect of design research that uses design as a means to address a research question (Frankel & Racine, 2010). There are several reasons to turn to research through design in order to produce a design framework. For starters, it helps ensure that design theory remains in touch with the realities of design—a framework is more likely to be useful if designers are directly involved in its formulation. Also, there is a strong affinity between the nature and purposes of design frameworks and the epistemological agenda of research through design, which Peter Downton describes as a “a vehicle for acquiring and shaping knowing that assists in future design activities” (Downton, 2003, quoted in Frankel & Racine, 2010). That is the epistemological stance that this research has taken. I adopted a contrarian design approach in order to problematize the manner in which video game design textbooks frame the design of video game interfaces and used this experience to propose a set of concepts that describe alternative ways in which game interfaces can foster, enhance and even create gameplay. The next chapter will be dedicated to the project that served as the practical basis for this contribution. I will introduce the project, demonstrate its relevance to the problematique, and provide an overview of its development to serve as context for what is to follow.

CHAPTER 2: MOVING BEYOND STANDARD INTERFACE DESIGN

2.1 INTRODUCING *HAMMURABI*

Hammurabi is the first game to be produced as part of *Interfaces Subjectives*, a FRQSC-funded research creation project that “re-frames virtual interactive characters as ‘subjective interfaces’ with the purpose of highlighting original affordances for interactive storytelling through conversation” (Lessard & Arsenault, 2016). The relevance of subjective interfaces to this research operates on many levels. For starters, re-framing the characters as interfaces broadens the definition of what a video game interface is beyond the domain of input peripherals and GUI widgets (in a manner not unlike Jørgensen’s gameworld interfaces). Lessard and Arsenault also underline the specificity of video game interfaces when they remark that “much in contrast to HCI experts, the bulk of the game designer’s work is to actually complicate interaction in interesting ways” (2016). Moreover, they directly invoke the concepts of uncertainty and information flow as they refer to subjective interfaces as “a diegetically grounded way to design a game’s information flow” whose value as game mechanics would be decided by whether or not they afforded “original and interesting ways to make interaction uncertain” (Lessard & Arsenault, 2016). As the first subjective interface game, *Hammurabi* embodies a break from established interface design conventions. It is also a game that explicitly problematizes information flow between the player and the game as the very foundation of its gameplay. Both of these characteristics make it a good case study for investigating possibilities that are marginalized by current theories.

I was involved in *Hammurabi*’s development for the entirety of its duration. Due to my having more development experience than the other members of the design team, I was informally cast

in a lead design role during pre-production. As such, I am well positioned to articulate how *Hammurabi* and its development reflect on the current state of the video game design literature as it pertains to interface design. In this chapter, I will provide a summary of the game's eighteen months development cycle while highlighting elements that are especially relevant to this research. I will start by outlining the project's original design proposition: remaking management sim forerunner *Hamurabi*³ (Dyment, 1968) and explain how we had to alter the original game's design in order to accommodate the project's ambitions and constraints. As I explained in Chapter 1, this part of the project was recovered through interviews conducted with other members of the Lablablab team. Moving on, I will describe the main challenges that the team had to overcome throughout the game's development as well as the manner in which we chose to approach them. In doing so, I will examine several prototypes that were retrieved from the project's git repository and show how successive iterations converge toward the final game. Finally, I will conclude with an assessment of the game's reception.

2.2 CONCEPTUALIZATION: REMAKING *HAMURABI*

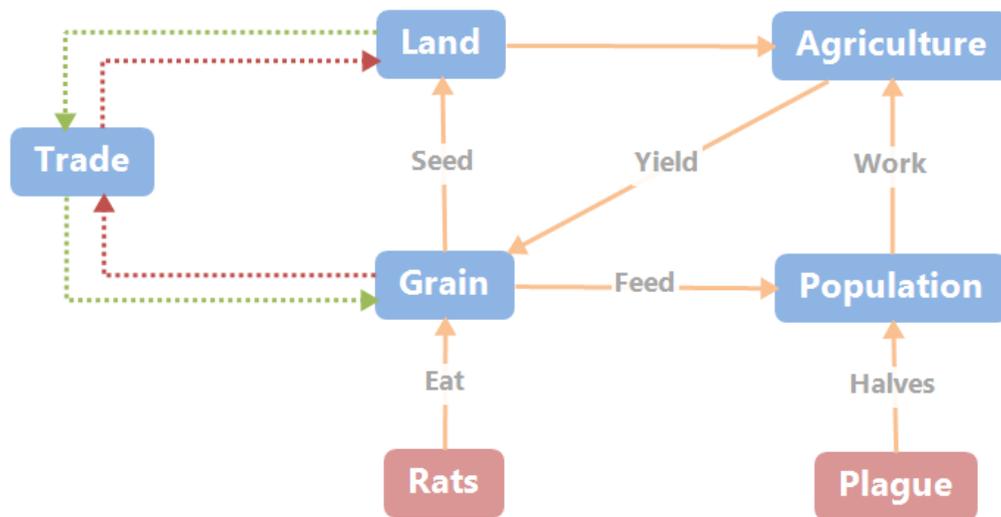
In their 2016 paper that outlined the project's theoretical foundations, Lessard & Arseneault propose three scenarios that could be compatible with subjective interfaces: meddling, investigation and leadership. While all three of these scenarios leverage a lack of information that makes the result of the player's actions uncertain, the leadership scenario gave us the opportunity to work from an existing game: *Hamurabi* also known as *The Sumer Game* (Dyment, 1968). This was deemed advantageous for several reasons. First, it allowed us to pour our efforts into the design of the subjective interface itself. Since we were going to be working on the first subjective interface game using largely untested technology, it felt prudent to keep

³ The title of the original game was spelled with a single 'm'—*Hamurabi*. However when developing the remake, the *Lablablab* team opted for the more commonly accepted spelling of *Hammuruabi*.

the scope as small as possible⁴. Working from an existing game also afforded the possibility of comparing the game to its predecessor, which provided added context for evaluating the impact of our design decisions. Secondly, we were specifically interested in seeing what form *Hammurabi*'s subjective interface would take and how its presence would affect the rest of the game's design.

Hamurabi (hereafter referred to as *Hammurabi 1968* for the sake of clarity) had a very rudimentary interface. It was originally developed for the PDP-8 minicomputer whose sole output device consisted of a teletype printer, meaning that the game data was printed at runtime rather than displayed on screen. Although many later versions ran on machines equipped with computer terminals, the game's output remained entirely textual. Its premise is simple: the player is cast in the role of Hammurabi, king of Babylon from 1792 to 1750 BCE, and must manage the resources of the city in order to survive their ten year reign while scoring the highest possible amount of points. The game includes three key resources: population, land and grain. Grain is used to feed the population, plant crops and acquire new land which is used to plant crops and can be sold back for grain. That being said, the maximum amount of land that the player can seed is also limited by population, since a unit of population can only work ten acres of land per turn. Population is also tied to the game's losing condition, as it the game ends once it falls below a certain threshold. The city can be struck by plagues, which dramatically reduce its population. Rats also periodically raid the player's grain stores, although this is more of an annoyance rather than a full blown crisis. The player must correctly invests the resources at their disposal in order to grow Babylon's reserves and keep a large enough population to be safe from plagues.

⁴ The original *Hammurabi* is a small game whose entire code can fit in a single (finely printed) page.



In order to adapt the game to the needs of the project, we first had to reverse engineer its rules and mechanics. We did this by keeping track of how the different resource values changed during gameplay. This allowed us to build an approximate model of the game’s internal economy which was then refined over several play sessions. This ‘black-box testing’ approach certainly would not work with most contemporary management games, but luckily for us *Hammurabi 1968* was simple enough that we managed to accurately reproduce its runtime behavior in a matter of days⁵. Once we had an accurate picture of the game’s underlying architecture, we had to tackle our first design challenge: what should *Hammurabi*’s subjective interface look like? What should it do? Moreover, we were already aware that the design of a subjective interface was heavily dependent on how the underlying game handled characters, challenge and information, but how those very qualities would react to the addition of a subjective interface was still unclear. Would rules and mechanics need to be altered? And if so, which ones and how?

⁵ This work was completed before the team became aware the the entirety of *Hammurabi 1968*’s source code was available online. That being said, the game was simple enough that reverse engineering it by interpreting its source code (as opposed to the black-box testing approach) would not have resulted in a deeper insight or a meaningful amount of time saved.

2.2.1 Viziers

In *Hammurabi 1968*, the line at the top of the game's output at the beginning of the turn reads "Hammurabi: I beg to report to you," The game thus characterizes itself as an ongoing exchange between Hammurabi and a royal advisor whose job is to report on last turn's events. As pre-production on the *Hammurabi* remake got underway (hereafter referred to as simply *Hammurabi*), the design team was keenly aware that we would need to broaden the distance between the player and the game state in order to leave room for the subjective interface. In order to achieve this, the team picked up on the idea of the royal advisor and expanded on it. Rather than having the player be directly in control of the game's mechanics, several 'viziers' were added to serve as semi-autonomous intermediaries that carried out the player's orders and reported back on the results. Each vizier thus functioned as a subjective interface in its own right, with their individual capacities, dispositions and temperaments determining the degree to which their actions conformed to the player's instructions as well as the thoroughness and accuracy of their reports. The decision was made to add more than one vizier because doing so would allow more opportunity to experiment with characterization.

2.2.2 Ministries

Adding the viziers meant that a system would need to be put in place that allowed the player to assign them tasks. To that end, the individual operations that the player could perform in the original game, namely (1) allocating grain to plant crops, (2) feed the population and (3) invest in the land trade, were grouped into three related ministries of agriculture, welfare, and warfare. The game's main loop was starting to take shape, as the turn would begin with advisors reporting on the result of their actions, whereupon the player would assign each vizier to a ministry and divide up their available resources amongst them.



The addition of the ministries had the noticeable side effect of compartmentalizing the game state in a manner that reduced the amount of information available to the player. At this point in pre-production, the game only included two viziers, meaning that the player would have to tend to one of the ministries themselves and leave the rest up to the viziers. By rendering certain parts of the game state inaccessible to the player, their attention was effectively being funneled towards the viziers' dialog as the player had no other way of knowing what was going on in the other ministries. Consequently, the player's ability to interpret the vizier's feedback by building up a mental model of their motivations and concerns rapidly emerged as a central feature of *Hammurabi's* gameplay.

2.2.3 Approval

Since a large portion of the information that the player received was now susceptible to be deformed by the viziers' subjective view of the game world which made the game far trickier to play, the viziers' states, goals and capacity for independent action needed to be meaningfully

related to the game's outcome in order for this added resistance to be framed as a challenge rather than an annoyance. Therefore, the decision was made to add a fourth key resource to the game's internal economy: approval. As with the viziers themselves, this idea was not entirely new. The flavor text that accompanies *Hammurabi 1968*'s "game over" screen suggests that the player has been dethroned as a result of losing the people's favor. Once again, the team took this idea and expanded on it, elevating it from narrative salad dressing to what became one of the game's core mechanics. From that point on, each character (including the player) had an approval score that would rise and fall in response to the character's actions. If at any point the player's approval score fell below that of one of the viziers, the latter could, if so inclined, overthrow the player which would result in a game over. This added a layer of political intrigue to a gameplay that primarily revolved around resource management, which resulted in a game that had far more interesting stories to tell. From the player's perspective, the game became about walking the fine line between having a prosperous kingdom and having prosperous viziers where having both or neither made for a short playthrough.

2.2.4 Crises

The final major change that was made to *Hammurabi 1968*'s design during pre-production was the addition of a crises system. As with viziers and approval, the team expanded on an idea that was present in the original game. That is, the plague event where a sizable percentage of the city's population was instantly wiped out by disease. The team liked how plagues added a element of tension to the otherwise monotonous resource management gameplay. On the other hand, it was found that plagues incentivized a degenerate play pattern where the player would wait for a plague to strike, then sell a portion of the land they could no longer work to dramatically inflate their grain stores. Since a less populous city had a proportionally lower grain upkeep, players that abused this mechanic could easily coast on their large grain stores until the end of the game's ten turns length. This was a rather obvious and heavily contrived optimal

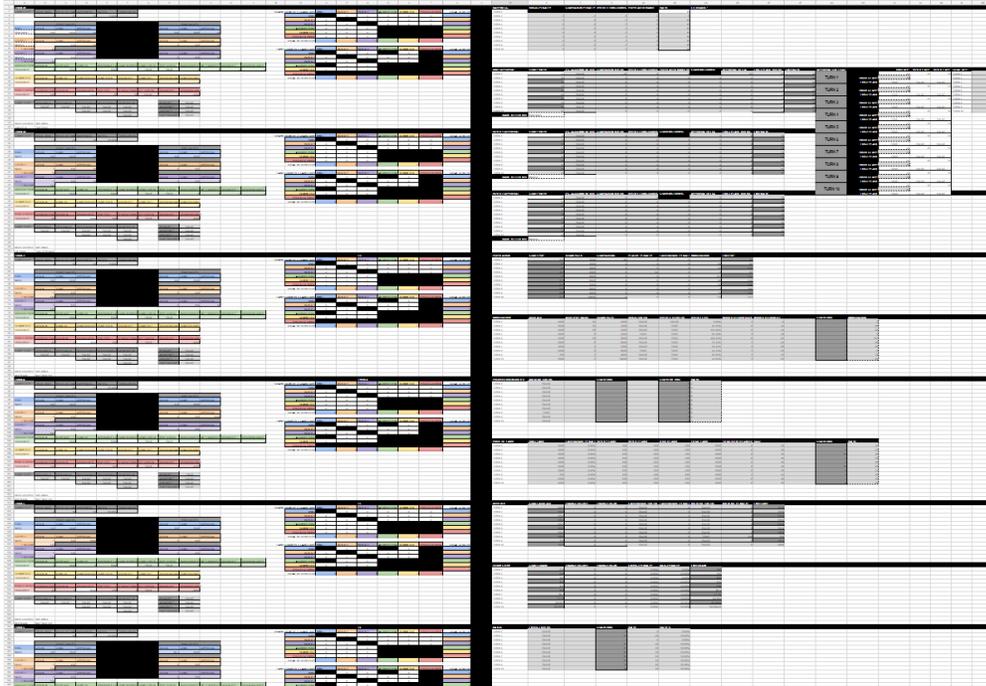
strategy that rendered the rest of the game's challenges moot. To counter this, the team added several types crisis that targeted all of the game's resources rather than only population, making stockpiling one or the other more of a gamble than a sure investment.

In summary, the design of the original game was significantly altered in order to align with the project's goals. Viziers were added to serve as intermediaries between the player and the game and information flow was restricted to make it necessary for the player to go through the viziers in order to interact with the game state. The approval score was introduced in order to make the viziers' states and motivations more relevant to the game's outcome, and new crises types were added to eliminate the problematic play patterns that were caused by plagues. As a side note, it is interesting to note that none of those additions were entirely absent from the original game. At this point in pre-production, the game's overall architecture was already well-defined. The next challenge in line was the design of the subjective interfaces themselves. How exactly would the viziers influence the game's outcome? What actions would they be able to take? How would they communicate with the player? And, perhaps more critically, how would their own state be reflected in the quality of the information that they passed on to the player?

2.3 PROTOTYPING & ROLEPLAYING

In order to decide on how best to answer these question, the team turned to roleplaying as a prototyping technique. This is not without precedent: researchers at UCSC's *Expressive Intelligence Studio* devised a "computationally-assisted live-acted prototype" in order to "guide content creation" for another game (Ryan et al., 2015). In *Hammurabi*'s case, two members of the design team would play as the viziers, while a third (or an outside participant) would play as the king. This presented several advantages. Roleplaying is well-suited to the design of games

that have a strong fictional component or systemic games that attempt to simulate the behavior of fictional characters. It allowed the team to rapidly experiment with characterization and delivery without getting bogged down with implementation details. In addition, the exploratory nature of roleplaying was a good fit for the project's rather experimental direction. We were able to rapidly get a sense of how and when the game became fun and explore these possibilities in a fast and improvisational way, which was especially helpful at a time when the team was still in the process of apprehending the value of subjective interfaces as game mechanics. Since *Hammurabi's* turn-based gameplay did not rely on an elaborate visual representation, roleplaying did not require a notable imaginative leap on the part of the participants. Most interactions between the participants consisted of dialog, which could easily be recorded and compiled for the design team to later parse through. That being said, even at this very early stage the game involved enough number crunching and partitioning of information to make roleplaying sessions grind down to a snail's pace. Some form of software support was needed in order to speed things up. The game was thus implemented in *Google Spreadsheet*, and the design team used its collaborative features to turn it into a multiplayer platform while the conversations between the king and the viziers were held (and archived) in a dedicated *Slack* channel.



The screen capture above features only a small fraction of the spreadsheet prototype. Showing it in its entirety would be rather impractical as it consists of seven sheets with the 'master sheet' (which hosts all of the game's background computations) stretching to 561 rows.

-  **odile** 4:22 PM
My lord ! It seems that it's a good time to buy some land ! We have an excellent price this year : 17 bushels. A good opportunity to take some advantage.
-  **chrisvtan** 4:22 PM
grrr, those cheap bastards
we're selling some
-  **etiennebl** 4:23 PM
At this price, my lord?
-  **odile** 4:23 PM
I demand your pardon my lord? You want to sell some land at this price?
-  **etiennebl** 4:23 PM
Are we in such dire straits already?
-  **odile** 4:24 PM
Your population is growing, you should reconsider the opportunity to get a bigger property and a little more power on the other villages. (edited)
-  **chrisvtan** 4:24 PM
i could starve your precious peasants
but you said it yourself, better to not
-  **odile** 4:25 PM
Exactly, I'm glad you are so wise my lord.
-  **etiennebl** 4:26 PM
[@odile](#): My, what a dreadful sense of humour our king has.

The screen capture above shows an excerpt from the archived roleplay sessions.

After approximately two months of weekly role-playing sessions, the team was able to isolate several emerging trends. First of all, it quickly became apparent that overly agreeable viziers made for rather boring and needlessly tedious playthroughs. The inefficiencies of subjective interfaces are not inherently meaningful or interesting—they require conflict and intrigue to come

alive. The team realized that the viziers were at their most effective when they were trying to further their personal agendas, especially when those agendas pitted them against one another—or against the player. Some of the game's most memorable moments occurred when the viziers were referencing one another in their exchanges with the player, offering backhanded compliments or indulging in outright slander. This development was especially interesting from an interface design perspective since it opened up a new information channel through which the subjective interfaces could reveal critical game state information without breaking character. For example, if vizier A was on the verge of securing enough popular support to stage a coup, vizier B would tip the player off, giving them enough time to undo vizier A's plot.

By tracking what conversations between viziers and the king usually revolved around, the team found that the viziers' dialog was limited to a handful of topics, namely the particular operations that were conducted in the ministries that they were tasked to oversee, the current state of the city as whole, the vizier's own state (including their disposition towards the player, the other vizier or their current occupation), and the state of the other vizier. However since the exact details of the viziers' character were never exposed to the player, the game frequently crossed the line from opacity to imperviousness as the player simply had no idea of what to expect from the viziers. In order to remedy this, the viziers were given strong personality traits that were clearly expressed in their dialog. If a vizier was curt but generally forthright in manner, the player was less worried about them being untruthful in their reports. Likewise, giving the viziers' 'tells' or singular speech patterns that indicated they were possibly lying or incorrect was an effective way to keep players in the loop.

The team also experimented with personality modeling techniques in a bid to make the viziers' behavior more system driven. However, those techniques were invariably found to be too reductive, especially as the game began to rely more heavily on characterization. The final

game includes a vestige of these efforts in the form of a half implemented 'traits' system where viziers would react to the player's actions depending on where they landed relative to their personal values. Although it initially showed some promise, the system was never finalized because its purpose and complexities did not translate to the player's experience in a meaningful way. What's more, as the team became more comfortable working with subjective interfaces, we recognized of the importance of properly managing the game's underlying complexity (this idea will be discussed at greater length in Chapter 3). In the case of *Hammurabi*, it was found that increasing complexity made it harder and harder for the player to know why things were happening. This is better explained in terms of information flow. Subjective interfaces rely on deliberately restricting information flow in a manner that engages the player's problem-solving skills. When the output of too many systems is funneled into the interface's limited 'information bandwidth', it becomes saturated and the feedback it provides goes from merely noisy (which is expected for a subjective interface) to outright unintelligible. This indicates that subjective interfaces may work better when the amount of information flowing from the game to the player is low, and one way to way to achieve this is to limit the game's underlying complexity⁶. Many of the systems that were developed during *Hammurabi*'s prototyping phase did not make it to the final implementation, and many of those that did were later scoped down as resources were funneled towards core features.

In summary, prototyping and roleplaying allowed us to refine our design problem into to the following : how can we reveal just enough to give the player a sense that there is more going on under the hood than what appearances suggest, all the while keeping them guessing and engaged? And how do we do this while establishing and maintaining characterization since any illusion of subjectivity breaks down the moment subjective interfaces stop being believable as

⁶ Interestingly, another counter-intuitive design solution in light of general game design manuals that usually advocate for increased complexity.

characters? As we shall see in the next section, the solution that the team produced was heavily contingent on the particular technology that was used to support the game's final software implementation.

2.4 IMPLEMENTATION

Game development is very much a multi-pronged battle and to cover the entirety of *Hammurabi's* development would extend beyond the scope of this thesis. This next section will focus on the areas that are most relevant to the subject matter of this research, namely the design of *Hammurabi's* interface. Both its subjective and graphical components will be covered in that order since it mirrors the chronology of the game's development. As the team moved away from the prototypes and into development proper, *Hammurabi's* design was significantly scoped down. Players could no longer directly control one of the ministries. This had major implications for information flow since denying the player direct access to the ministries effectively removed an important information channel from the game. Rather than choose what part of the game state they wanted to expose for the next turn, players now shuffled viziers across the ministries and cross-checked their fragmentary and often contradictory reports in order to assemble an approximate picture of the game state. If we think of subjective interfaces as characterized by a twofold nature as both characters and interfaces, this accentuated their role as interfaces. From that point on, characterization came as a second to more pressing concerns of playability. As we will see shortly, this is reflected in the design of the system that generated the viziers' dialog.

2.4.1 Hammurabi's Subjective Interface

2.4.1.1 Procedural Dialog Generation with *Expressionist*

The viziers' dialog needed to represent their individual perspective on the game state, but *Hammurabi* is a dynamic and unpredictable game. In order for the viziers' to be able to respond adequately to all the situations that could arise during gameplay, they required an equally dynamic (and unpredictable) procedural dialog generation system. This was a pivotal moment in the project since the technical challenges of dialog generation ate up the bulk of the team's resources for the rest of the game's development. In order to give the viziers voices, we established a partnership with UC Santa Cruz researcher James Ryan who was working on an experimental text generation tool called *Expressionist*. More precisely, *Expressionist* is a tool for authoring context-free grammars or CFG's. CFG's are simple computational structures that allow the generation of a large number of sentences (upwards of 10^{30} in some of the more elaborate grammars) from a relatively limited set of rules and textual components. In practice, CFG's function like phrasal templates or sentences that contain one or more empty spaces that users may later substitute with words or data. In the case of CFG's, empty spaces can be filled with other phrasal templates, and this process can be repeated an arbitrary amount of times. As an example, let us imagine a CFG that is designed to talk about the properties of certain animals. The initial template might look like "The [animal] is [property]" where [animal] and [property] have built in rules for substitution that determine what they can be replaced with in the final expression. The output of this grammar might be something like "The cat is black" or "The dog is hungry." What distinguishes *Expressionist* from similar tools such as Kate Compton's *Tracery* (2014) is a feature that allows the user to markup certain textual components (such as [animal] and [property] in the example above) with metadata or 'tags'. To go back to our example, this would allow us to specify what kind of animal we want to talk about. By tagging the word 'cat' with a distinct piece of metadata, we can then write a separate piece of software

that looks at all the possible outputs of the grammar and select only the ones that talk about cats⁷. In other words, the possibility space of *Expressionist*'s grammars is effectively searchable. This provided a degree of authorial control and flexibility that no other tool could match.

That being said, *Expressionist* was always meant to be the front end of a fully functional text generation system. It allows users to author CFG's but as I said above, a second piece of software—called a 'productionist'—is needed that takes in the grammars and processes them in order to produce actual sentences. Since this processing is highly application-specific, designing and implementing this productionist was the first technical challenge that *Hammurabi*'s developers had to overcome. From a high level design perspective, the productionist had to monitor key game state variables and search the CFG's that contained the viziers' dialog for content that matched the situation. However since the viziers were intended to be unreliable, several strategies were explored to give the productionist more flexibility in negotiating the match between the viziers' dialog and the game state configuration that it was meant to represent. Initially, the team settled on a solution where the productionist tolerated content that did not match the underlying game state so long as it did not result in an outright lie. This is best explained with an example. The game was designed so that the variable that described the state of Babylon's grain reserves could only be in one of three possible states at any one time: low, medium or high. In a situation where Babylon's reserves were low, the productionist would exclude content that referred to the grain reserves' other possible states (medium or low). The rest would be treated as normal, meaning that the vizier could talk about the city's low grain reserves... or really anything else. This was deemed to offer an acceptable compromise between serendipity and coherence. In practice however, the CFG authors rapidly

⁷ Admittedly, this is a gross simplification. For a more in-depth discussion of CFG's, *Expressionist* and how they were used in *Hammurabi* and other game projects, see Ryan et al. (2016) and Lessard et al. (2017).

converged towards an authoring style that made this feature more or less superfluous. The reasons were twofold: first, these behaviors made debugging and iterating exponentially harder.⁸ The limitations of *Expressionist's* authoring environment simply did not allow the authors to make efficient use of this more probabilistic feature. Secondly, it made the game too difficult to play. The viziers' feedback was often eccentric to the point of arbitrariness and players tended to disregard it entirely. In the final game, the vizier's unreliability is almost entirely realized through the authors' writing.

2.4.1.2 From Generativity to Modularity

This adjustment maximized the amount of control that individual authors could exert on the feedback that the viziers would provide. They also meant that the final game would fall somewhat short of the team's original ambitions, at least in the extent to which it made use of the generative power of CFG's in producing the vizier's dialog. As Lessard *et al.* reported in a previous paper that described the CFG's authoring pipeline in more details, dialog generation in *Hammurabi* steadily moved away from generativity as development progressed (Lessard *et al.*, 2017). The final implementation is better described in terms of modularity, or "the use of discrete units, called *modules*, to assemble larger structures" (Grinblat, 2017, original emphasis). This is because the amount of time and resources that the team would have needed to dedicate towards designing and implementing a convincing, system-driven characterization solution would have been out of proportion to its contribution to the player's experience.

The project's trajectory is symptomatic of the tension that is intrinsic to procedural content generation, that is the necessity to reach a compromise between authorial control (or author agency) and what we may call procedural agency. Half the work of a procedural content

⁸ This is an inescapable aspect of working with emerging or untested technology. In most cases, the optimal tools and techniques have yet to be invented.

generation system is managing the generator's output. In a large majority of cases, raw unfiltered outputs just won't do. As Kenny Bachus noted, such outputs rapidly begin to exhibit similarities that betray the generator's built-in patterns (Backus, 2017). Worse, homogeneous and pattern-riddled content rapidly stops being interesting to players. As Bachus puts it, "we want generators to make something a human would make, or something we didn't expect would be made [...] We want to be pleasantly surprised" (Backus, 2017). This is especially true of procedurally generated dialog where homogeneous outputs can damage the characters' believability. In order to make a convincing procedural dialog generator for, say, allowing the player to have passing conversations with the inhabitants of a small town, a relatively simple approach would be to author a set of parameters that allowed for a very wide range of possible outcomes and to pair it with a set of checks and heuristics designed to break up patterns and similarities. Different projects however call for varying levels of authorial and procedural agency. A subjective interface needs to be believable as a character whilst allowing the player to play the game. This means that in practice the system has to generate dialog that is at once an accurate reflection of the game state *and* something that the player would expect the character to say. In other words, it needs to say just the right thing in just the right way. This amount of constraints as well as the impracticality of working at a higher level of abstraction combined to steer the project further and further away from generativity.

2.4.1.3 CFG Templates

Since the design team included three CFG authors, it was decided that each author would be responsible for one of the game's three viziers. This allowed us to explore different authoring approaches as we were still familiarizing ourselves with *Expressionist*. However as development progressed, the limitations of the available authoring tools became more and more apparent. The lack of adequate debugging tools made it difficult for individual authors to track down and

resolve problems in the grammars. More often than not, solving these problems required the attention of two or more members of the team. Likewise, the limitations of *Expressionist's* authoring environment (namely the absence of key usability features such as an undo command, a search function or the ability to organize symbols into a hierarchical structure) made it hard to iterate on existing grammars, especially as they ramped up in complexity. As a result, when characters would start to behave erratically, the authors often chose to scrap the faulty grammars and start over. This state of things quickly led to the development of a set of standard authorial practices that eventually coalesced into a CFG template that served as the basis for every new grammar. On top of making it easier for the authors to work together to resolve technical issues since they no longer had to learn each other's authoring logic, the template also allowed them to get new grammars up to speed quicker. That being said, the presence of the template is reflected in the system's outputs in that most of them loosely adhere to a structure where the viziers introduce themselves, give their report, and conclude with a statement that hints at their current moods and preferences. While this could be seen as an undesirable generative artifact, the particulars of *Hammurabi's* setting help mitigate this issue—after all, it stands to reason that the communication between a king and his advisors would be somewhat formulaic. The grammars were also designed to leverage this pattern in a manner that strengthened characterization. For example, the reports of the vizier called Tamraz rigorously stick to the template which befits his stringent personality, whereas the other more iconoclastic viziers take much more liberties with it. Below is an example of how the viziers' reports can differ in form as well as in tone even when they reference the same game state configuration. In this particular case, the viziers are reporting on the state of the welfare ministry after a particularly difficult year that saw a locust plague eat its way through the kingdom's grain reserves, resulting in massive starvations.

Doumbaf	Tamraz	Cassandra
<i>“Long story short, a fat bunch of ‘em starved, pardon the pun. I’d tell you how many but... y’a know, numbers, me, that’s really never gonna happen.”</i>	<i>“Feeding people is going to be difficult since locusts ate our grain. Despite the crisis, I fed them. Just not enough... The Gods are doing this because you are a prick.”</i>	<i>“Your majesty, our population isn’t going to make it if we keep neglecting it like this.”</i>

As we can see, the viziers’ outputs differ a great deal. Doumbaf does not even mention the locusts since they did not affect him directly. Both Tamraz and Cassandra blame the player for the kingdom’s misfortunes, but Tamraz at least offers a hint of how the events of the last turn might factor into the player’s future decisions. Cassandra is uncharacteristically laconic, but this is an aspect of her whimsical character.

In the final implementation, each vizier consists of several CFG’s that govern their decision-making and dialog. The system achieves its goal of making interaction uncertain in several ways. The viziers introduce an element of ‘delegated input’ where the player’s commands are entered in the form of instructions that the viziers then carry out to the best of their abilities. The uncertainty lies in how the viziers’ performance is influenced by their personal preferences and competences which can fluctuate as the game progresses. Those values are never exposed to the player who instead must intuit them by paying careful attention to the manner in which the viziers report on the result of their actions. When the project was still in pre-production, the team had hoped that the subjective interface would carry all of the game’s feedback. As we shall see in the next section, it soon became clear that several key variables would need to be exposed in the game’s graphical interface because players needed a baseline of reliable information in order to properly contextualize the feedback provided by the viziers⁹. As a consequence, the

⁹ The final game features a ‘voice of the people’ mechanic that was added late in development specifically to give players more data to contrast against the feedback provided by the viziers.

nature of the information carried by the vizier's feedback evolved as development went along. Rather than serving as an interface to the game's internal economy as per the project's original premise, the vizier's became interfaces to their own state as well as (perhaps most importantly) to each other's state. This particular configuration was selected because it made the game playable and engaging, but it also meant that a significant portion of the game's information flow would need to be handled by the game's graphical user interface.

2.4.2 Hammurabi's Graphical User Interface

This where the contrarian aspect of *Hammurabi's* design truly came to the fore. Until then, the inefficiencies introduced by the subjective interfaces could have been offset by a sufficiently transparent and functional graphical user interface. However, the design team went very much in the opposite direction. As we shall see shortly, every successive iteration of *Hammurabi's* GUI *reduced* clarity and transparency rather than magnify them. The final GUI configuration features a drag and drop operation that allows the player to assign the viziers to the ministries, a button to advance to the next turn as well as a turn indicator. These were left relatively untouched throughout development because they were deemed essential to the game's playability. The last two features however—the *resources gauges* that tell the player how much resources they have and the *resource sliders* that allow them to allocate resources to each of the ministries—were designed and redesigned many times over, with each passing iteration made to be less functional than the previous one. As we shall see, the final solutions both skirt the line between dysfunctional and nonfunctional.

2.4.2.1 Resource Gauges

Perhaps ironically, the first Unity Engine implementation of the game had an interface not at all dissimilar to that of the *Google Spreadsheet* prototype. As we can see in figure 1, the player

allocated resources to ministries by inputting numerical values and the current values for population, acreage and grain stores were exposed.

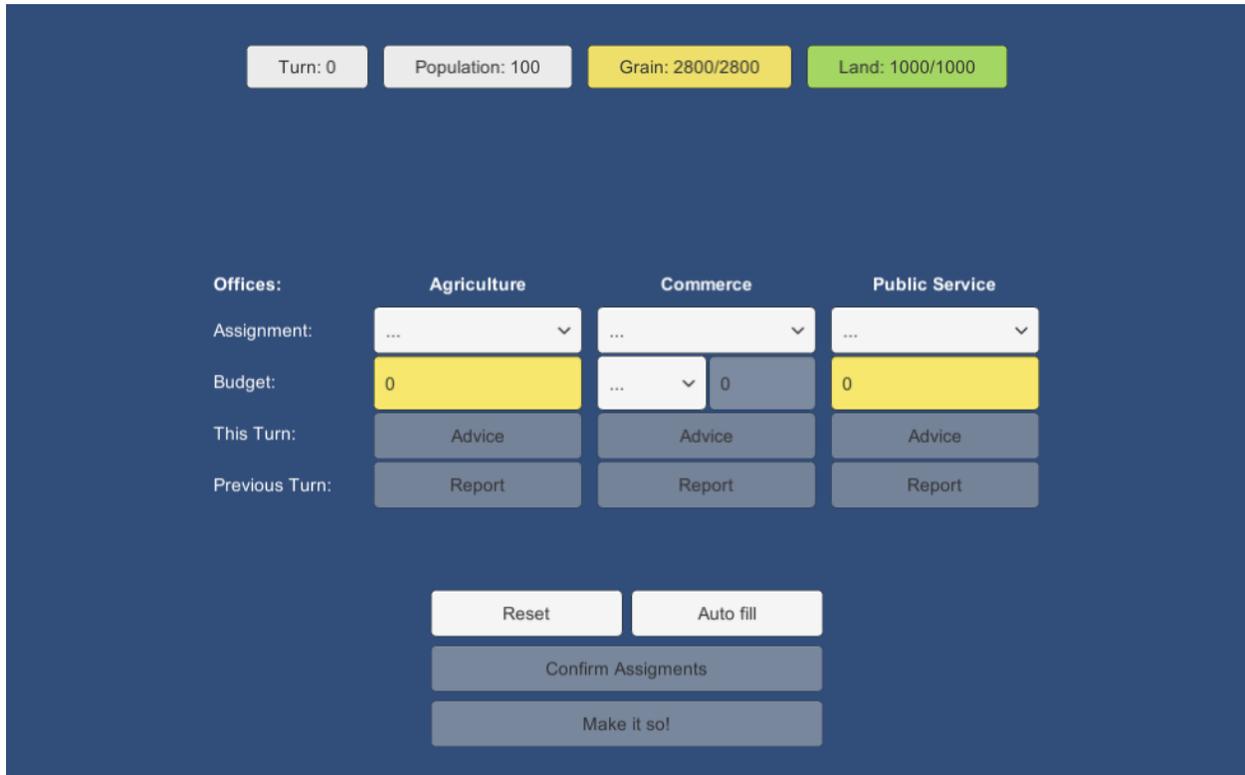


Figure 1

After testing, it became apparent that giving the player access to those values made the game a little too transparent in a way that damaged the overall experience. For example, let us imagine a scenario where the player allocates 1000 grain to the agriculture ministries on turn 1. On turn 2, they get 2000 grain back as crop yields and thus understand that the agriculture mechanic works by multiplying the allocated grain value by an integer. If the player then allocates 1500 and gets back 2700, she can deduce that the vizier must have pocketed the missing grain. In other words, since *Hammurabi's* internal economy is so simple, exposing key game state values as raw numbers meant that the player could easily reconstruct the game's internal operations by performing rudimentary mental calculations, thus making the subjective interfaces transparent and thus superfluous. It also created frustration in players who chafed at being unable to punish the corrupt viziers. Clearly, the interface needed to be more opaque in order for the game to

work as intended. Certainty was not something we could afford to provide players with. That being said, some key game state variables still needed to be exposed in order for players to be able to make reasonably informed decisions. For starters, they needed to be able to gauge how much population remained in the city, since this value was tied to the game’s primary losing condition. Likewise, they needed to have a way to tell if the grain stores were abnormally low because this would require them to alter their decision-making. Since exposing these values directly had proven detrimental, they would need to be represented through a visual approximation. In figure 2, we can see an early iteration of this idea where the grain and population values are expressed as stacks of identical icons.

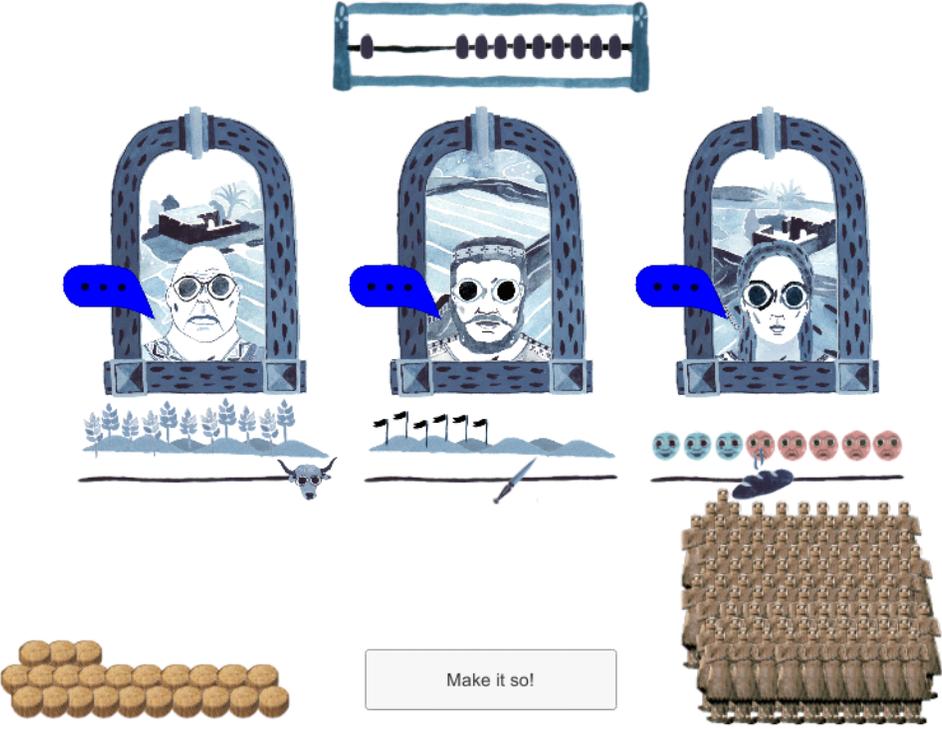


Figure 2

While this was a step up from raw numbers in terms of opacity, clever players could still intuit how much of a given resource each individual icon was worth and work out the game’s operations from there. Moreover, the identical icons were deemed to be damaging to the game’s visual appeal. In figure 3—which features the game’s final GUI assets—the grain and

population gauges are composed of icons of different sizes and shapes that toggle on or off as the amount of resources that the player has at their disposal fluctuates.

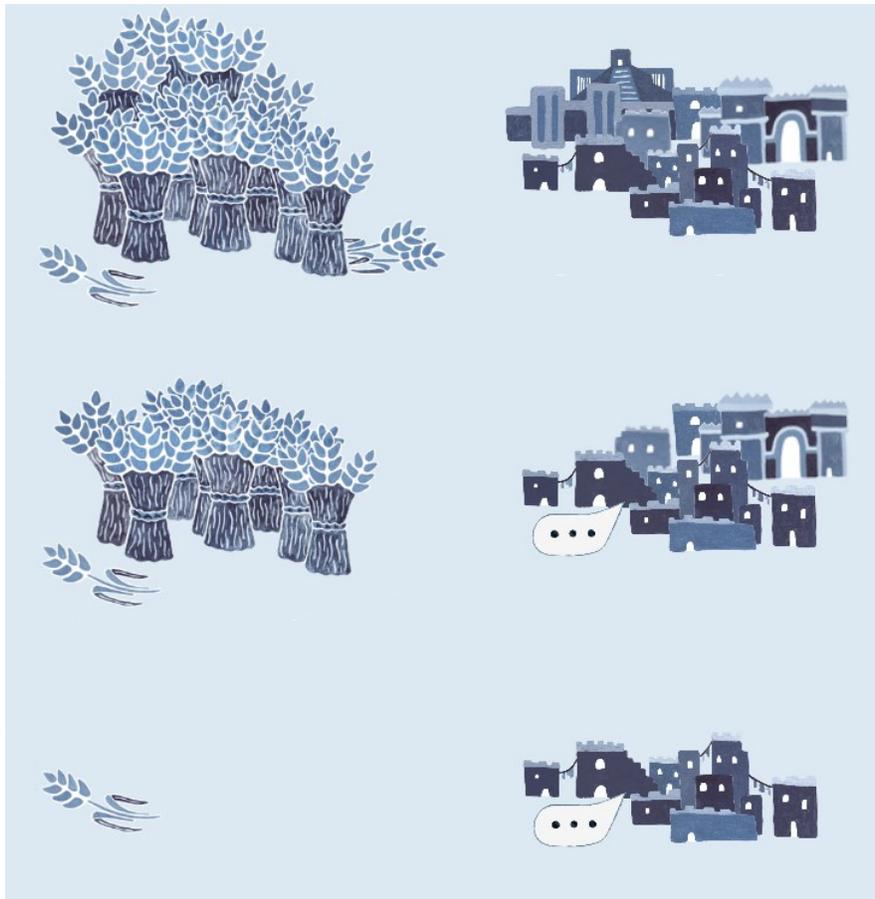


Figure 3

The heterogeneous assemblages of icons were designed to prevent the player from approximating the actual amounts of grain or population they had left and instead only allow them to evaluate whether their stocks were threading downwards or not. Since they represent a significant step down in both the clarity and the quality of the feedback provided, these gauges are anathema to standard UI design practice. Yet they are integral to how *Hammurabi* realizes its intended gameplay.

2.4.2.2 Resource Sliders

As raw numbers were being phased out of the game's output, inputting resource budgets as numerical values became increasingly awkward. As we can see in figure 4, sliders were added to the interface relatively early in the implementation phase.

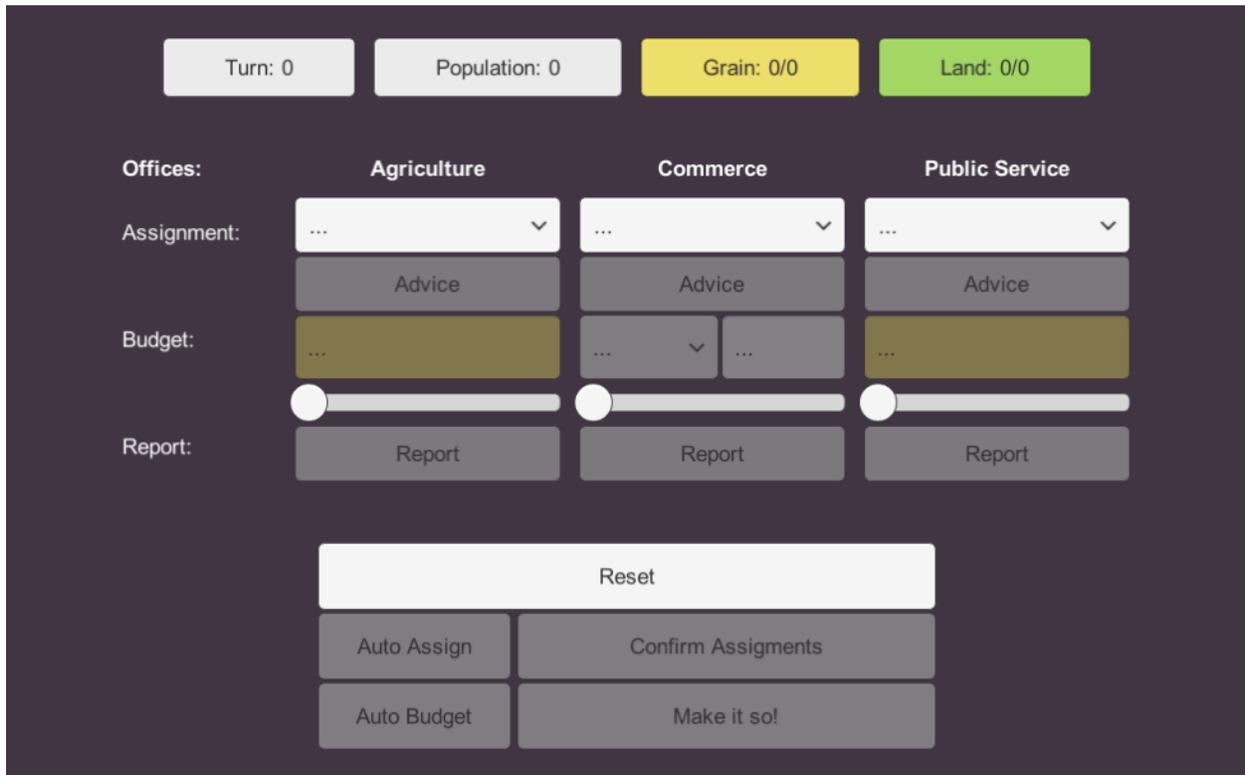


Figure 4

Although the actual numerical values were still displayed in the input fields, the team was already experimenting with more manual methods of input¹⁰. Figure 5 shows a later iteration of this particular input mechanic where the numerical values have been removed and the sliders fill up with the appropriate resource.

¹⁰Sliders also helped alleviate the computational burden that previous iterations of the interface placed upon players. For example, in order to allocate an adequate amount of grain to the welfare ministry, players had to multiply their current population value by the amount of grain that a single unit of population requires to survive until next turn. Consequently, players who were less proficient at mental calculations often had to play the game with a calculator in hand.

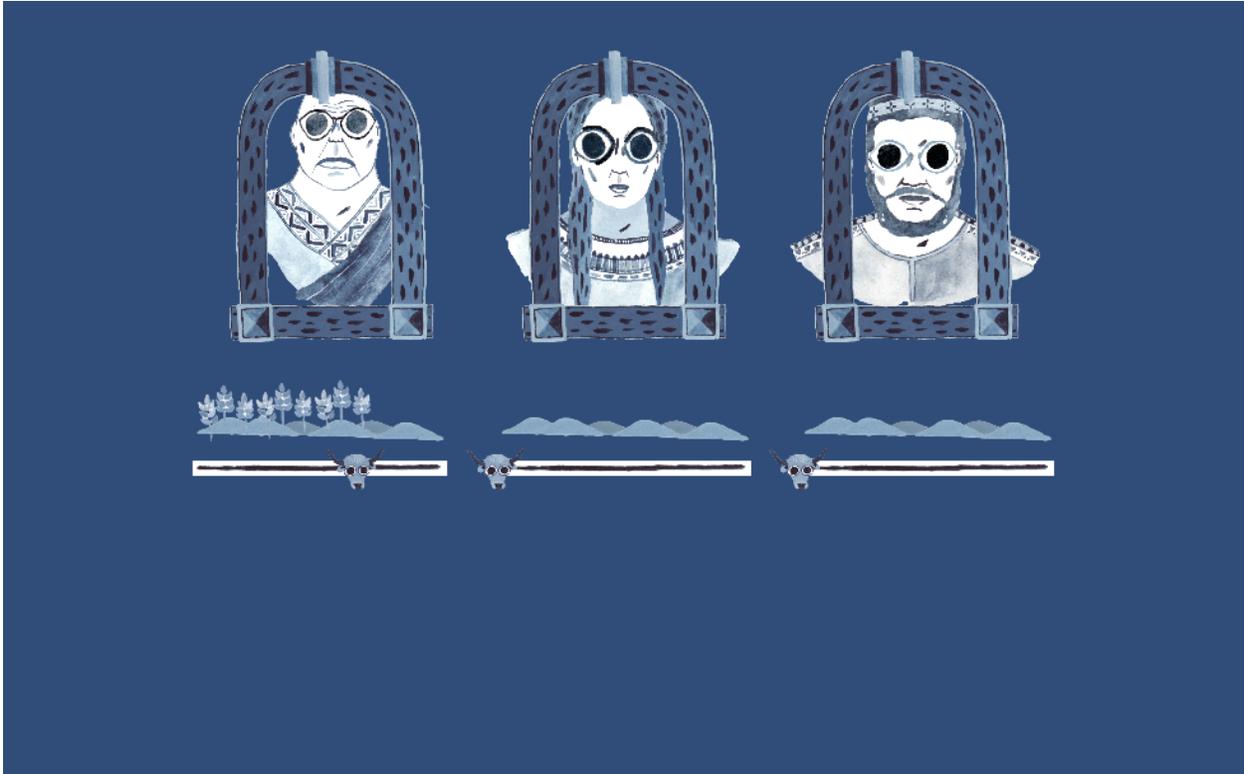


Figure 5

Later still in figure 6, the sliders work jointly with the grain gauge at the bottom left to allow the players to divide up the available grain amongst the three ministries (the grain icons disappear when a slider is pushed to the right, indicating that an undisclosed amount of grain has been allocated to the corresponding ministry).

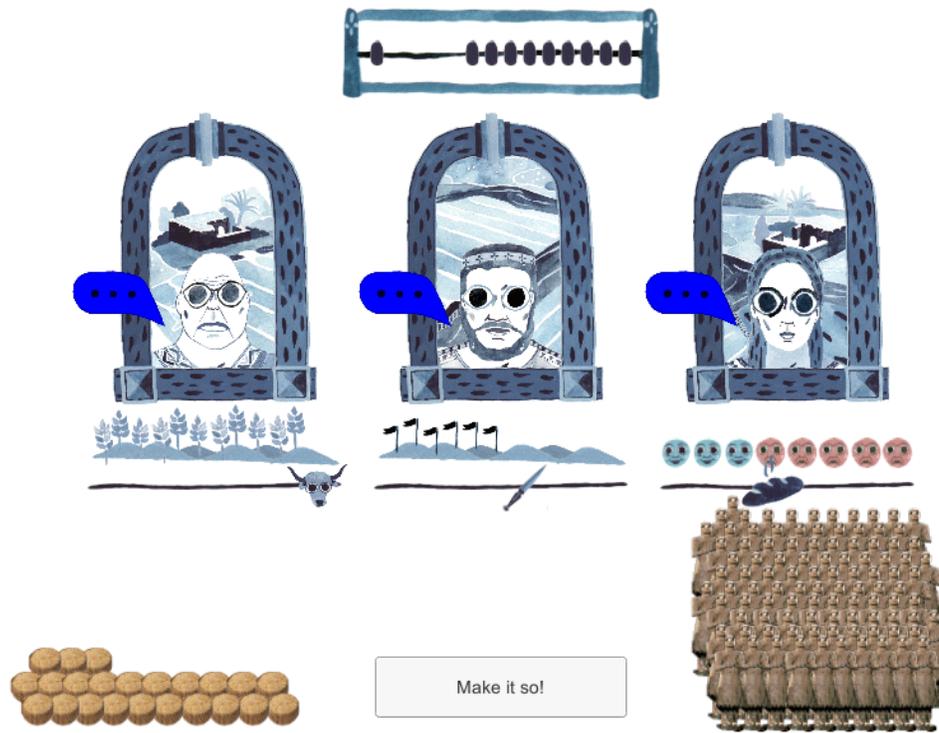


Figure 6

However, the ministries are more than mere resource drains. Each of them presents the player with the potential to either maintain or add to their current stocks of grain, land or population. In that respect, they function more like transactions where the player invests a certain amount of resources in order to achieve a desired result. In *Hammurabi 1968*, the player could estimate the result of a particular transaction by looking up the relevant game state variables and performing the necessary calculations in their head. Since those values are not exposed in the remake's interface, the design team needed to find another way to allow players to anticipate the outcome of their actions. This turned out to be a persistent and difficult problem. In the case of the agriculture ministry—certainly the most straightforward of the three—the solution was relatively simple. When the player allocates grain to agriculture, grain icons disappear from the grain gauge at the bottom left. As the amount of grain invested increases, 'ghost icons' (icons with their alpha channel set to semi-transparency) begin to peek up from behind the regular icons. These ghost icons represent the expected return on the player's investment. That is, the

amount of grain they should get back barring extraordinary circumstances or vizier meddling. This particular solution is on display in figure 7.

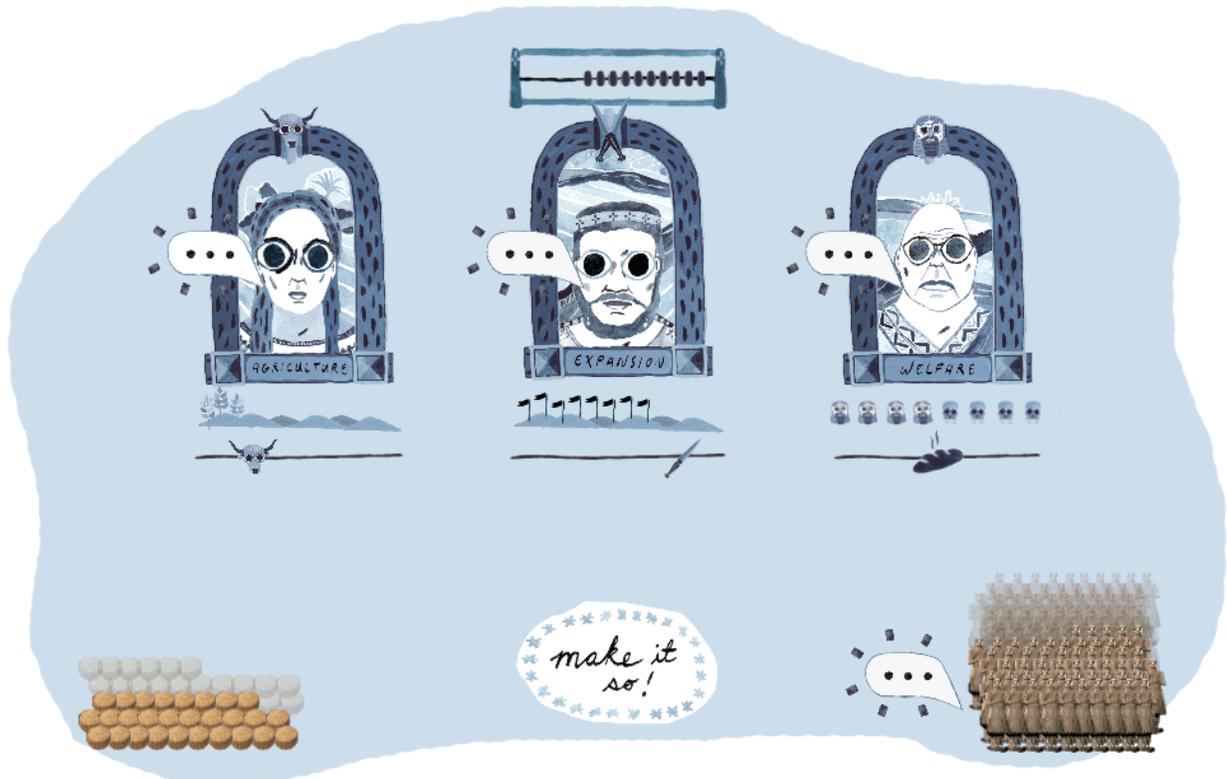


Figure 7

A similar solution was used for the welfare ministry, with the exception that the input slider was also modified to reflect the proportions of fed and unfed citizens given the amount of grain allotted (also in figure 7). The warfare ministry was a more complicated affair. The other two ministries had both an input slider and a resource gauge for the design team to work with, whereas the warfare ministry's input slider doubled as its gauge. Worse, the warfare transaction takes in not one but two separate resources (land and grain) and involves a conversion between the two¹¹. In the final game, the full length of the ministry slider represents all the land that exists in the game. The part on the left of the thumb is filled with flags and represents the portion of existing land that the player currently controls (see figure 7). By moving the thumb along the

¹¹ As a reminder, the player can perform two separate actions through the warfare ministry: they can either invade new land at the cost of grain or cede land in exchange for grain.

slider, the player can increase or decrease the amount of land they control. If the player slides the thumb to the left—therefore ceding land in exchange for grain—ghost icons begin to appear in the grain gauge that represent the amount of grain that the player can expect to receive on the next turn (again, barring any natural disaster or vizier meddling). By moving the thumb to the right, the player can attempt to claim new land under their rule. The larger the claim, the more grain icons begin to disappear from the grain gauge as grain is being diverted towards mounting the coming invasion. Admittedly, this solution is more ambiguous than the other two. That being said, its impact on the overall experience is lessened by the fact that the warfare ministry is less integral to the game's outcome than agriculture or welfare.

2.4.3 The Contrarian Interface

From the perspective of a standard user experience designer, *Hammurabi's* graphical user interface leaves much to be desired. The feedback that it provides is sparse and vague. Its methods of inputs are awkward and fail to foster an intuitive mapping of the user's intent over their available means of action. The result is a sense of remoteness, as if the system is actively trying to keep to user at a distance. This stands very much in contrast—or in contrary—to how video game interfaces are supposed to behave. This was a deliberate choice on the part of the design team, as was just discussed. The limitations of the interface from a standpoint of functionality were the result of intentional design, not an incapacity to make things clear to the player. The prototypes on display in the figures are unequivocal proof of this as each new iteration contains less unfiltered information than the last. The GUI was designed to funnel the player's attention towards the project's *raison d'être*: the viziers. Moreover, the game as a whole was designed to leverage the inefficiencies of its interface as sources of uncertainty and challenge. The remoteness of the interface helps reinforce this aesthetic by presenting the player with very little conclusive data to support their decision-making. Whether or not this challenge occasionally spills into frustration seems to depend on the degree to which individual

players are responsive to the game's central conceit, *i.e.* that players should attempt to control the behaviour of the system without being given the optimal tools and information to do so. As my colleague Christopher Tan (a fellow member of the *Hammurabi* design team) put it, "*Hammurabi* a game about dealing with necessary evils and making the best out of a bad situation." What differentiates it from other games that share a similar premise is that it specifically problematizes interaction at its most surface level to achieve its expressive aim rather than having it emerge out of the interactions of complex game systems.

2.5 RECEPTION

Almost a year has passed since *Hammurabi*'s release which gives us ample distance to reflect on how it was received by general audiences. *Hammurabi* was made available for free on online hosting platforms such as *Newgrounds* as well as on the Lablablab's own website in early December 2017. It was also exhibited at several events including the *Montreal Independent Game Festival*, the *Montreal Expo Gaming Arcade*, the *Wordplay* festival in Toronto, and The Museum of Waterloo. On the show floors, player response was hit or miss as *Hammurabi* struggled to stand out among the multitude of games on display. It should however be noted that the players who took an interest in the game often stayed at the booth for long periods of time (upwards of thirty minutes). Moreover, *Hammurabi* fared much better in online venues where it made the front page of *Newgrounds* and received generally favorable reviews. In both cases, the game was able to appeal to a niche but enthusiastic audience which makes it more than a scholarly exercise in contrarian design¹². *Hammurabi* is a prime example of how video game interfaces constrain information flow to create intrigue, suspense, conflict and challenge¹³. As I argued in chapter 1, the video game design literature currently lacks the necessary

¹² At the moment of writing, *Hammurabi* was played more than 9000 times on NewGrounds, 39 players chose it as a "favorite", 230 voted for an average score of 3.79/5 stars.

¹³ Other titles such as Kitfox Games' *Shrouded Isle* (also on display at the 2017 *Montreal Expo Gaming Arcade*) also use the opacity of their interface to a similar effect, albeit not to the same emphatic extent.

aesthetic and formal vocabulary to provide for a meaningful discussion of this aspect of interface design. As such, the next chapter of this thesis will feature my attempt at addressing this issue as I will be putting forward a way of thinking about interface design that makes salient the principles and considerations that underlie the design of *Hammurabi* and other games of its ilk.

CHAPTER 3: TOWARDS AN EXPANDED UNDERSTANDING OF GAME INTERFACES

3.1 BEATING *HAMMURABI*

As I explained in chapter 1, the primary aim of this research was to address the discrepancy between how video game interfaces actually function and how they are described within video game design textbooks. I argued that our current definitions of video game interfaces were too narrow and that our design framework, which stresses that interfaces should maximize transparency and control, failed to account for how interfaces can bolster, enhance and even create gameplay, particularly from the standpoint of information flow. I hypothesized that purposively throttling information flow in a manner that inhibited the player's agency could result in compelling gameplay experiences, even if doing so cut squarely against the grain of standard design practice. The creative part of my research allowed me to put this idea to the test. *Hammurabi's* interface actively resists the player's attempt to exert their will upon the game, to the extent that it constitutes the game's main source of challenge. Over repeated playthrough, the player aggregates a more stable and actionable model¹⁴ of the game's black-boxed contents (*i.e.* the rules that govern its internal economy and the data that makes up the viziers' state). Once the player has reached the point where they can correctly interpret the viziers' feedback and anticipate their behavior, the challenge is void and it is highly likely that the player will prevail. In other words, to beat *Hammurabi* is to master its interface. Central to this *Hammurabi* effect is the design of the game's contrarian interface as it must obscure the game state while walking the fine line between challenge and frustration. Rather than transparency and control, we propose to think in terms of *opacity*. Although all video game interfaces will eventually

¹⁴ This model is not necessarily accurate, but it is stable in the sense that the game's runtime behavior is less likely to cause the player to revise it and actionable in that it increases the player's agency over the game state.

become functionally transparent to the player given an arbitrarily long interaction window, an opaque interface will take longer to do so. In that sense, the interface's opacity level really describes the amount of time it takes before the player can see through it. In the next section, I will lay out some of the core concepts and strategies that were elaborated during *Hammurabi's* development and which proved helpful in approaching the design of its opaque interface.

3.2 DESIGNING FOR OPACITY

Opaque interfaces are awkward to design within current design frameworks since they frame them as inelegant or failed solutions. The purpose of this section is to propose a high level design vocabulary that will assist designers in exploring solutions that lie outside the bounds of standard practice. I chose to present them in the form of a series of design lenses (in the spirit of Jesse Schell's book *Art of Game Design*) because this format lends itself well to how these concepts are meant to be used. They are meant as windows into unfamiliar or counter-intuitive possibilities, to be used and discarded as the situation dictates. Each lens consist of an explanation of the relevant concepts and a set of questions that help designers bring their inquiries into focus. The lenses offer a coarse-grained picture of the dependencies and constraints that underlie the design of opaque interfaces which can then be refined and tailored to the priorities of individual projects as the iterative process runs its course. For added context, each lens is followed up by examples that show how these concepts reflect on *Hammurabi's* design.

Lens #1 : Availability of Game State Data

This is undoubtedly the most straightforward of the lenses presented in this chapter and the one that is most readily associated with the idea of an opaque interface. This makes it a good

place to start from. Quite simply, the availability of game state data describes the portion of the game state that the player can 'see'. Naturally, most games do not expose the entirety of their runtime data since doing so would result in information overload where the player is unable to process all of the data that they are presented with (in that sense, it could be said that maximizing the availability of game state data can actually *increase* opacity). The key difference between an opaque interface and one that subscribes to standard usability practices is that the former overtly presents the player with obstacles for them to overcome. The first step is thus to establish what is known to the player and what is not since is doing will dictate which problems they will have to solve. Once that has been decided, designers can add in ways for the player to work around the obstacles. Here are are some strategies for tackling this problem.

- Make an inventory of all the data that makes up the state of the game at the lowest level of abstraction. What information *must* be made available in order for the game to be playable (e.g. data that relates to win/lose conditions or that is essential to operating the game's core mechanics)?
- What does the player want to know? Can this information be denied to them and if so, how does this impact the gameplay?
- How much of the remaining data can be omitted or left up to the player's interpretation? How does this affect the player's perception of events?

As was discussed in chapter 2, *Hammurabi* does not allow the player direct access to its in-game economy. All inputs and outputs flow through the game's AI-controlled character, which establishes them as the game's primary obstacle. This was effective in focusing the player's attention on their interactions with the characters, as exposing too much of the game state would have rendered the subjective interfaces superfluous. Other games also partition their

game state data in a manner that shapes their gameplay. The ‘fog of war’ mechanic where the player’s vision does not extend past the line of sight of the units under their control is perhaps the most literal implementation of this idea, and it has been nearly ubiquitous in both real-time and turn-based strategy games for several decades. Likewise, racing games will typically force the player to alternate between looking forward to negotiate turns and avoid hazards and backwards to track other racers. Finally, in *Five Nights at Freddy’s* (Scott Cawthon, 2014) evil robots converge on the player’s location who must track their movements and position using security cameras. The robots generally don’t move when the player watches them, but the player can only look through one camera at once. The game effectively leverages these gaps in the player’s perception of the game state to create a feeling of suspense and dread. In all of these case, the game takes a piece of information that the player very much wants to know and denies it to them, which shapes the gameplay in profound ways.

Lens #2 : Information Flow

Information flow describes the manner in which information that pertains to the game state is funneled through the interface’s input and output channels, where the input channels allow the player to input commands while the output channels carry the system’s feedback. The amounts of information that these channels can convey determine the interface’s input and output *bandwidths*¹⁵. If the amount of information that is exchanged between the player and the game exceeds this bandwidth, information flow becomes *saturated* and the game becomes unplayable. While full on saturation is never a good thing, an interface that approaches saturation makes it more difficult for the player to *map* their actions onto the

¹⁵ To use a commonplace example, a character’s health bar is an output channel that exposes a single quantity within the game state. It allows the player to keep track of the character’s health, but tells them nothing about the reason(s) why the health value is changing. Therefore, the health bar is limited in the amount of information that it can convey. If the interactions between the health system and the rest of the game are made too numerous or complex, a single health bar can quickly become saturated.

system's reactions, thus increasing its opacity level. This can be achieved in several ways. For example, lowering the input bandwidth (for example, by removing input channels) can *decrease* the game's opacity since it makes it easier for the player to probe the system through a black-box testing approach. Conversely, a lower amount of output channels will typically *increase* opacity since it increases the odds that different input will yield a similar answer, making it harder for the player to infer causal relationship. When deciding what kind of information flow would benefit a game the most, it is helpful to ask oneself the following questions.

- What inputs/outputs channels are built into the game's interface?
- How many things is the game doing in response to the player's actions, and how does this amount compare to the interface's output bandwidth? Is the interface at or approaching the point of saturation and is this desirable here?
- In the case that saturation does support the game's intended experience, which input channels could be added to the interface in order to make the mapping of inputs and outputs less linear? Which output channels could be removed, merged or otherwise obstructed in order to make harder for the player to perceive and/or interpret the system's reactions?

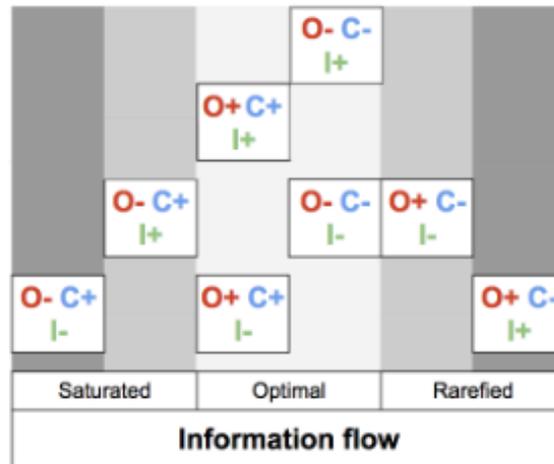
In *Hammurabi's* case, the interface features a very narrow input bandwidth. The player's available actions are limited to assigning resources and viziers to each of the three ministries and pressing the turn button. The amount of output channels is comparatively high as the interface presents the players with resource gauges, vizier dialog and the people's voice. However, the game output bandwidth is not commensurate with its underlying complexity, which causes some output channels to saturate. This is most evident in the resource gauges which, as was discussed in chapter 2, convey more information than their design can reliably handle

(especially the warfare gauge). Consequently, *Hammurabi* would sit toward the leftmost boundary of the above diagram. Both its input and its output bandwidth are narrow, especially in relation to its underlying complexity, which creates a labored information flow. Saturation thus factors into the high opacity of *Hammurabi*'s interface, although much of it comes from other sources. This illustrates how opacity should be approached in a manner that is similar to game balance where the end result is the product of the interactions of several moving parts. In that sense, we can think of the design lenses that are featured in this chapter as design levers that should be fine-tuned to create the desired effect.

However, if information flow can exceed the interface's transmission capacity, it stands to reason that it could also fall short of it, thus creating a situation where information flow becomes *rarefied*. To the best of my knowledge, no such design exists. *Hammurabi* achieves its effect mainly through a combination of saturation and low quality feedback (see *Lens #3: Quality of Feedback*, p.56). The affordances of rarefied information flow in video game interfaces have thus yet to be explored—if they exist at all. It may turn out to be a purely diagnostic concept, in the sense that its purpose is to diagnose faulty game interfaces. However, many designers and researchers today would argue the same about saturation and opacity in general, even though they are central to *Hammurabi*'s design. Future research may thus reveal how rarefaction, much like saturation, can also be leveraged as a source of opacity.

The output bandwidth of an interface can only be meaningfully evaluated in relation to the amount of information that it is funneled through it. The interface's information flow is therefore determined by the interactions of three key parameters: the interface's input bandwidth, its output bandwidth, and the complexity of the underlying game systems (see *Lens #4: Underlying Complexity*, p.57). The many ways in which these parameters can interact to produce different effects are represented visually in the diagram below.

		Output Bandwidth		Underlying Complexity	
		High	Low	High	
Input Bandwidth	High	O+ I+	O- I+	O+ C+ I+	O- C+ I+
	Low	O+ I-	O- I-	O+ C+ I-	O- C+ I-
Underlying Complexity	Low	O+ C- I+	O- C- I+		
		O+ C- I-	O- C- I-		



The values above are approximative and meant to help designers apprehend and evaluate the options that are available to them in terms of information flow in game interfaces. Examples of saturated or rarefied information flow are a rarity in existing games because their applications for game design have yet to be explored in depth. Ocelot Society's *Event[0]* (2016) is one such rare case as its premise is somewhat similar to *Hammurabi's*. The player is cast as an astronaut investigating the fate of a derelict spacecraft. In order to do, they must befriend the ship's onboard computer called Kaizen 85. The player's interaction with Kaizen are text-based, and the AI's personality is responsive to the player's attitudes and actions. Underlying complexity is thus quite high, but the only means of interacting with Kaizen is the conversation interface, which is more limited that initial appearances may suggest. Kaizen's dialog is plentiful and varied¹⁶, but it will only respond to a specific set of commands from the player. The system's output bandwidth is thus commensurate with its high underlying complexity, but the player's ability to poke and prod at it is comparatively low, resulting in a somewhat labored information flow. Referring to the chart above, we could describe the information flow of Kaizen's conversation interface as a type O+, C+, I-, still within the optimal zone, yet exhibiting a degree of resistance to interaction. The player is never able to know with certainty what Kaizen is

¹⁶ In a 2016 interview for eurogamer, the game's developers have alleged that Kaizen's dialog generation system could produce over two million lines of dialog.

thinking or why. Of course, this works to the game's advantage as preserving the mystery as to Kaizen's motives and identity is integral to the game's plot.

For a more mainstream example, From Software's *Dark Souls* (2011) is infamous for its refusal to expose its character building mechanics to the player. *Dark Souls'* class leveling and equipment systems are both fairly elaborate, but the player is given very little cues as to the impact of their decisions, especially in the mid to long term. The optimal builds have since been theorycrafted made available on online community platforms. However at the time of the game's release, it was not uncommon for players to find themselves locked into a subpar build that made the game nigh unbeatable. In terms of information flow, *Dark Souls's* character building system can be described as a type O-, C+, I+, where the game's output bandwidth is decidedly not commensurate with its complexity, resulting in a saturated information flow. *Dark Souls* gets away with this by styling itself as an antagonistic, often punishing experience where the opacity of the interface is just one of the many ways in which the game challenges the player.

Lens #3 : Quality of Feedback

Feedback that is either unreliable or difficult to for the player to interpret or parse through will increase opacity. Unreliable feedback is typically relayed by semi-autonomous agents that are either error-prone or deliberately misleading. Feedback can be difficult to interpret when it is communicated through indirect means (for example, the player can witness the consequence of a non-player character's action rather than the action itself) or when multiple feedback sources are in contradiction. Likewise feedback that features a low signal to noise ratio is more difficult to parse, meaning that it is harder for the player to isolate the important part of the message.

- Could feedback be relayed through non-player characters and if so, how might their

characterization affect the quality or the content of the feedback?

- Which gameplay events could be implied rather than displayed?
- How could different feedback sources be made to convey divergent perspectives on gameplay events?
- Which feedback sources could be made to carry more information than necessary?

In Sid Meier's *Civilization V* (Firaxis, 2010) some AI leaders will attempt to placate the player with declarations of friendships. However should the player's military strength fall too low, these same leaders will pounce on the player's undefended territories without hesitation or regard for past declarations. That being said, the player is not totally left out of the loop: by observing the movements of the AI Leader's armies, the player can get a clue as to their current disposition toward their neighbors. This is an example of how feedback that is both indirect and misleading can enhance diplomacy dynamics in strategy games by adding a layer of intrigue to the mechanical transactions between factions. Other examples of indirect feedback include Kitfox's *Shrouded Isle*, where the characters have hidden personality traits which the player must uncover by observing the characters' behavior. In both cases, the player is denied a piece of information that is vital to the game's outcome, and must instead infer it by looking at how this information might become manifest elsewhere in the game world. Another approach is to dial down the feedback signal to noise ratio in a manner that echoes a common game design technique where the player is presented with a large set of elements and is then asked to isolate or select the ones that are most important or meaningful (3909 LLC's *Papers, Please* (2013) uses this principle in its core mechanic¹⁷). Parsing through the game's feedback then requires a non-trivial effort which the game can leverage as a source of challenge. *Hammurabi's*

¹⁷ The player is cast as a border crossing officer in the fictional country of Arstotzka and must spot missing or falsified information amidst the paperwork of potential immigrants before deciding whether or not they should be allowed into the country.

feedback combines all of these approaches. The game features multiple feedback sources that carry inaccurate, misleading and contradicting information. The viziers' dialog is often laden with unnecessary verbiage that obscures the heart of the matter and several gameplay events have to be deduced from observation, such as the viziers plotting to overthrow the player.

Lens #4 : Underlying Complexity

While it has far reaching implications for the entirety of a game's design and not just that of its interface, complexity is perhaps the single most important variable to keep track of when designing an opaque interface. As discussed in chapter 2 as well as in *Lens #2 : Information Flow* (see p.54), high underlying complexity tends to make opaque interfaces unstable since it can quickly saturate their limited information bandwidth and make the game all but unplayable. In general, it is more prudent to keep it at a minimum. However, that isn't to say that system complexity is necessarily something to be avoided. In many cases, a system that reveals itself to be less complex than what it suggests is met with disappointment as it fails to live up to the player's expectations. This is what new media scholar Noah Wardrip-Fruin has called the *ELIZA* effect (2009). Wardrip-Fruin also coined a term that describes the opposite scenario: the *Talespin* effect, where high underlying complexity is not reflected in the system's output, thus creating a situation where the player is not allowed to interact with the most interesting parts of the game's systems (2009). In both cases, the degree to which the player is allowed to experience the complexity of the game system is determined by the extent to which the player is able to 'see' through the game's interface. Complexity and opacity are thus locked in a delicate balancing act where the manner in which the latter allows the player to perceive the former will shape the players' expectations as to the game's procedural contents. The *ELIZA* effects illustrates how the nature of these expectations and the extent to which the game fulfills them should be handled with care, as they can have a profound influence the

player's experience.

- Are the channels of interaction commensurate to the possible gauging of complexity by the player? I.e. will the player ever be able to make sense of the system's workings?
- Does the game include systems that generate an amount of data that is disproportionate to their contribution to the player's experience? If so, can they be simplified, replaced or removed?
- What expectations does the game as a whole create in players in terms of the responsiveness, variety and flexibility of its run-time behaviour? Can the game's systems possibly meet these expectations in a satisfying manner? And if not, what measures can be taken to remedy the situation?

For most games, the relationship between perceived complexity and actual system complexity is hard to evaluate since their source code and detailed design documents are rarely made public. That being said, there is a generalized tendency to use clever design to make systems appear more intelligent, more autonomous and more complex than they actually are. The ghosts in *Pac-Man* (Namco, 1980) are a textbook example of this practice as their simple pathing logic manages to give the impression that the ghosts are working together to corner the player¹⁸, while in actuality individual ghosts are never aware of each other's position. *Pac-Man's* example shows how low underlying complexity does not necessarily result in an *ELIZA* effect. In most cases, players seem perfectly willing to forgive some measure of hand-waving on the designers' part, so long as the gameplay is sound. Perhaps because of the medium's performance-sensitive nature, game designers and developers have found ways to achieve success by doing more with less. Correspondingly, occurrences of Wardrip-Fruin's *Talespin* effect are rare in

¹⁸ For an in depth discussion of the logic that underlies the movements of *Pac-Man's* ghost, see Maré, 2018: <https://dev.to/code2bits/pac-man-patterns--ghost-movement-strategy-pattern-1k1a>

video games because normal game development tend to filter them out. However the meeting of player perceptions, expectations and reality can play out in many other ways. Wardrip-Fruin theorizes a *Sim City* effect which emphasizes the role of the graphical user interface in presenting the user with an “overt metaphor” for the game’s systems (2009). This metaphor serves as an approximate model for the game’s procedural contents which is then refined through trial and error. Whereas the ELIZA effect culminates in breakdown and collapse, the *Sim City* effect arcs upward as the player moves from experimentation to discovery and finally self-expression.

Hammurabi can be described as a variation on that idea. *Sim City* (Maxis, 1989) and *Hammurabi* both use their interface to provide the user with a baseline model of the system internals. However, they take the player on very different journeys. While *Sim City* was designed to get users excited about learning how to assemble complex virtual machines, *Hammurabi* styles itself as a game of competition between human and machine. And whereas *Sim City*’s arc runs from experimentation to discovery and self-expression, *Hammurabi*’s takes the shape of an uphill struggle where the odds of victory progressively shift in the player’s favour as the opacity of the interface gradually dissolves. In both cases, the complexity of the game’s systems and the opacity of its interface are tuned to allow the player to experience the former in a manner that supports the game’s intended experience. *Sim City* uses minimal opacity and high underlying complexity to facilitate learning, while *Hammurabi* uses high opacity and low underlying complexity to create challenge and allow for the eventuality of triumph. If *Hammurabi*’s systems were even more complex than they already are, the game might be too opaque to be enjoyable. Several of *Hammurabi*’s systems were simplified or scrapped in order to achieve this effect, such as the personality traits system, the random events system and a mechanic that represented the happiness level of Babylon’s population. As discussed in Chapter 2, these systems introduced extraneous complexity that was damaging to the overall

experience. That being said, the members of the development that were interviewed in preparation for the writing of this thesis were in agreement that the game would have benefitted from being streamlined further. Had the team become fully appreciative of the relationship between complexity and opacity sooner in the game's development, the final game may have turned out very differently.

Lens #5: Duration of the Interaction Window

Like complexity, choosing the duration of the interaction window will have repercussions across every aspect of a game's design. That being said, it is of special consequence to opacity because of the latter's time-dependent nature. Since opacity describes the amount of time it takes for the interface to become functionally transparent to the player, then the manner in which it relates to the duration of a typical play session will influence how the game's opacity is interpreted by the player. A low ratio of duration to opacity may incentivize replayability, especially if underlying complexity is high since this increases the amount of content that the player can uncover over repeated attempts. However, this is only appropriate in games that leverage the opacity of their interface as their main source of challenge. In more narrative-driven games where the payoff is tied to the plot's denouement, a higher ratio may be more advisable (perhaps as high as 1:1 where the interface has become functionally transparent by the time the player has finished the game's story). In both cases, the relationship between opacity and duration can help bring the game's intended experience into focus.

- What is the average length of a play session?
- How much of the game's challenge rests in the opacity of its interface?
- How long should it take for the interface to become transparent? Should it take more than one play session?

From the beginning, *Hammurabi* was designed as a 'die and retry' game where the player's first few attempts almost inevitably end in abrupt defeat. As a result of this approach, the duration of the interaction window was set in the 5-10 minutes range, as short play sessions incentivize replayability and lessen the sting of defeat. This helped the team define the game's opacity level. Since we wanted players to be able to figure out the game before their interest waned, it was deemed that the interface should become functionally transparent after 30-40 minutes of play. In *Event[0]*, the opacity of the conversation interface is an accessory to the game's through-line, that is uncovering the mystery that underlies the game's plot. As such, it becomes functionally transparent relatively quickly, over one or two playthroughs. Seeing as much of the interest of repeated playthroughs of *Event[0]* rests in experiencing the many ways in which its narrative can unfold, this also works to the game's benefit since it allows the player to easily navigate their way to the game's alternate endings. For a more contrasted example, *Dark Souls'* character building system is so thoroughly opacified that players need to invest several play sessions, each one lasting several hours, in order to be able to make sense of it. In this case, the duration of the interaction window is very high (a single playthrough of *Dark Souls* can stretch over a hundred hours), and the opacity level of the character building follows suit and becomes a meaningful chapter in the player's struggle.

Lens #6 : Levels of Encapsulation

Video games typically comprise several interconnected interactive systems (inventory systems, internal economies, progression systems, etc.) The game's interface must mediate the player's interactions with all of these systems, but it does not have to do so in the same manner. *Hammurabi* is a single screen game, but most modern video games are not. Put another way, the interface's opacity level does not have to be *uniform*. Adding a level of

encapsulation means creating an opaque layer around a specific part of the game, with different levels of encapsulations having varying levels of opacity (we think of it as creating a nested instance of this entire conversation). Many game systems are often 'individually wrapped' in this way. The details of how procedural content generators and AI behaviour routines operate are rarely exposed to the player which must instead derive their current state by observing their outputs. Encapsulation can have many uses such as focusing the player's attention on specific parts of the game, avoiding information overload by filtering extraneous data or adding gameplay to otherwise mundane systems. It is not strictly a tool to increase the opacity level of the interface, but rather a way to focus it. That being said, the interface's overall opacity level seems to be at least partly determined by that of its individual components. As a result, adding many levels of encapsulation will generally increase the game's overall opacity level.

- Make an inventory of the key systems that make up the game. How does the player interact with each of the systems?
- Which of these interactions are central to the experience? Could some of those interactions be streamlined to bring the experience into sharper focus?
- Do these interactions allow the player to experience the possibilities of each system to a satisfying extent?

As I said, this lens does not dovetail with *Hammurabi's* static, single screen design and relatively simple architecture. Rather, its purpose is to illustrate how these concepts can be ported to other designs that do not share these characteristics. For example, we can imagine a stealth game inspired by 1990's spy-themed action thrillers where the protagonist has to contend with enemies on both sides of the story's central conflict (in the spirit of the *Mission Impossible* and *Bourne* franchises). In this imaginary game, the player must of course infiltrate

enemy strongholds while avoiding detection and direct confrontation as much as possible in order to complete their mission objectives. In this moment to moment gameplay, the player very much wants to know three things: where the guards are, what they are doing, and where they will be next. However this is a proper espionage fiction thriller, and so the player has no guarantee that their mission objectives are what they appear to be. They are after all an expendable asset caught in the machinations of history. Some of these objectives could impact the game world in a way that the player did not expect or intend. Some of them could lead the player into a trap. The player *wants* to know which objectives are righteous and how fulfilling them will impact the plot, but in order for the game to deliver on its premise, they must never know for sure.

So here we have two systems that need to be *opacified* in order for the game to work as intended: the one that controls the guards' behavior and the one that presents the player with mission objectives. These systems are not entirely uncoupled, but they stand quite apart in both the nature of their outputs and the level at which they contribute to the player's experience (for example, they are embedded in short-term and medium to long-term game loops, respectively). As such, the strategies that designers use to opacify them will be very different. For example, the game could give the player a series of environmental clues that hint at the guards' location, their routines and their level of alertness. In this particular case, the game world itself would stand as the primary interface between the player and the AI-controlled guards where the latter's state is communicated through a visual and iconic 'language' that the player learns to interpret as they become more familiar with the game's level design. As for the mission objectives, they could be delivered through mission handlers which would themselves be wrapped in something akin to a subjective interface, leaving it up to the player to assess the level of trustworthiness of their superiors and colleagues. Because of the large differences in the manner and the extent to which these systems are exposed, it makes the most sense to think of

them as belonging to different levels of encapsulation even though they are both examples of opacity in video game interface design.

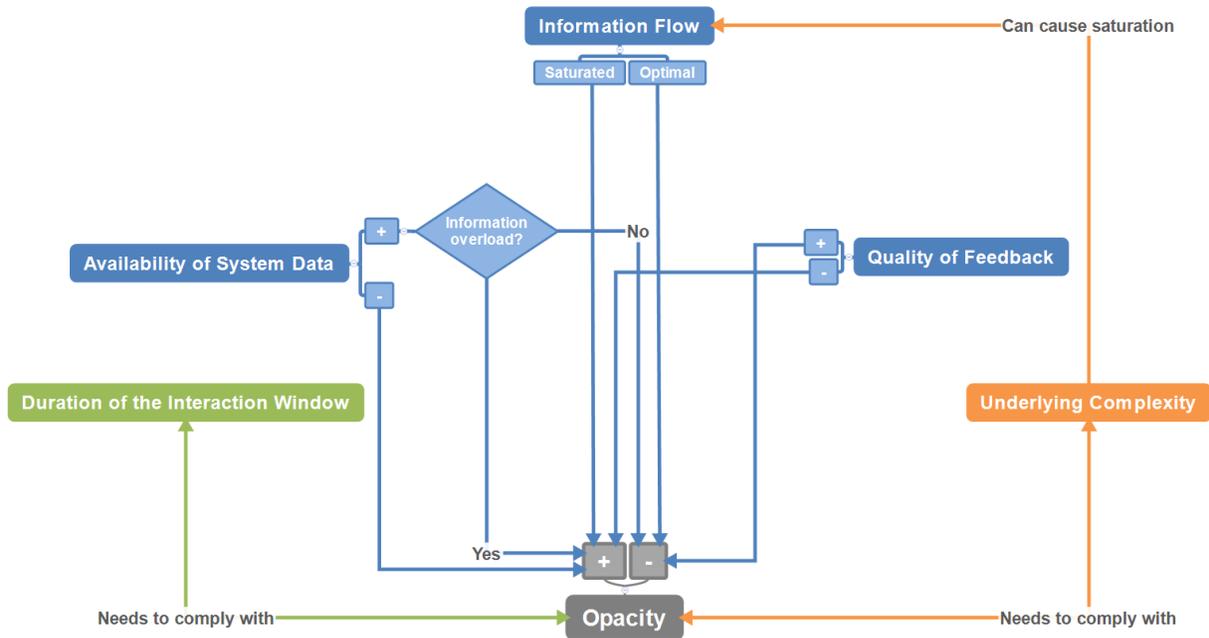
3.3 SUMMARY & DIAGRAM

The table below offers a summary of these concepts and shows how they can be arranged into a design method (although just like Jesse Schell's lenses, they don't have to be).

Availability of game state data	How much of the game state is the player able to access and of this amount, how much is directly exposed and how much must be inferred through indirect means?
Information flow	How can the interface's input and output channels be adjusted to create the type of information flow that would best support this inference process (saturated, optimal or rarefied)?
Quality of feedback	How can the quality of the game's feedback be adjusted to make this process this process adequately challenging?
Underlying complexity	Given the above, how much of the system's underlying complexity manages to translate to the player's experience?
Duration of the interaction window	Is the duration of the interaction window commensurate with the amount of time that the player would need in order to get the most out of the game?
Levels of encapsulation	This approach can applied to the game as a whole, particularly in the case of a single-screen design such as <i>Hammurabi</i> , but it can also be used to analyze the design of individual systems that make up a more complex game.

Although they can be packaged in this linear manner, we should not overlook the fact that these concepts are deeply interconnected. As I have stated before, they are better understood as members of an ecosystem where changes to one element can affect the others in dynamic and

unanticipated ways. The diagram below makes some of these connections explicit in the hope that future designers can better navigate this problem space.



Although opacity is first and foremost an interface design concern, it has repercussions across the entirety of a game’s design, from that of its rules and systems to its intended duration. Opacity, complexity and duration appear to be inextricably linked in ways that open interesting perspectives for future designs. For example, while it is true that too much complexity can cause an opaque interface to become saturated, the right amount of opacity can make a simple game appear more complex than it really is, provided that the interaction window is short enough that the player is not given enough time to see through the subterfuge. As I hinted earlier, this is an idea that *Hammurabi*’s design team would have been keen to explore given the right circumstances.

CHAPTER 4: CONCLUSION

As I argued in Chapter 1, the current conceptual framework of video game design is in a fragmented state where the concepts that we use to guide our perception of what makes up a game were borrowed from other disciplines and domains of knowledge, each with their own priorities and perspectives. When those concepts reach the limit of their applicability to video games, we are left with aporias in our theoretical discourse. This phenomenon was especially apparent in and around the topic of video game interfaces and their design where a large portion of the literature asserted that game interfaces should strive to maximize transparency and player agency. In making *Hammurabi*, the Lablablab team threw out every rule in the book and made a game that does not deviate from these conventions so much as it lampoons them. And yet, it works. It is not a perfect game, but it is very much a game with an audience to vouch for it. This success has allowed me to re-open the theoretical conversation surrounding game interfaces and offer a theoretical perspective that elucidates the design of *Hammurabi* and other games like it. It also validates the approach that this research has taken, where I used contrarian design to investigate the merits of hitherto negatively valued design possibilities. The future will see ample opportunities for other researchers to use a similar approach to shore up other gaps in our understanding of video games and their design, such as how players derive meaning from gameplay and the ever-looming gameplay/story problem.

At a more general level, the fact that this approach has proved productive speaks to the usefulness of games that call into question our definitions of what a game is or should be. These 'edge cases' afford us unique vantage points from which to criticize the assumptions that are built into our definitions. *Hammurabi* is an example of this, but not all edge cases need be the product of a research-creation effort. Edge cases abound on the video game marketplace where the incentive to innovate fuels constant experimentation. Walking simulators, for

example, have recently emerged as a successful if highly polemical game genre that blurs the line between gameplay and interactive storytelling¹⁹. The crux of the controversy surrounding walking simulators seems to center around the issue of whether or not they are in fact games. If *Hammurabi's* case is any indication, understanding how and why walking simulators succeed or fail as games may bring an expanded understanding of narrative gameplay.

¹⁹ *Dear Esther* (The Chinese Room, 2012[2008]), *Gone Home* (Fullbright, 2013) and *The Stanley Parable* (Galactic Cafe, 2013[2011]).

BIBLIOGRAPHY

Adams, Ernest W. 2010(2006). "Fundamentals of Game Design (2nd edition)". MIT Press, Cambridge, MA, USA.

Arsenault, Dominic. 2005. "Dark waters: spotlight on immersion". In Proceedings of GAMEON-NA. EUROSIS, Ghent, BE.

Backus, Kenny. 2017. "Managing Output". In T. Short and T. Adams (Eds.) Procedural Generation in Game Design. A K Peters/CRC Press, New York, NY, USA.

Bogost, Ian. 2008. "Persuasive Games: Windows and Mirror's Edge". Gamasutra. Retrieved on October 22, 2018, from www.gamasutra.com/view/feature/132283/persuasive_games_windows_and_.php

Chapman, Owen; Sawchuck, Kim. 2012. "Research-Creation: Intervention, Analysis and Family Resemblances". Canadian Journal of Communication, vol. 37, issue 1, april 2012. Retrieved on October 14, 2018, from <https://www.cjc-online.ca/index.php/journal/article/view/2489>

Compton, Kate; Kybartas, Ben; Mateas, Michael. 2015. "Tracery: An Author-Focused Generative Text Tool". In Proceedings of Social Media Fictions. Designing Stories for Community Engagement: 8th International Conference on Interactive Digital Storytelling, Copenhagen, Denmark.

Costikyan, Greg. 2013. "Uncertainty in Games". MIT Press, Cambridge, MA, USA.

DiSalvo, Carl. 2012. "Adversarial Design". MIT Press, Cambridge, MA, USA.

Downton, Peter. 2003. "Design Research". RMIT University Press, Melbourne, AU.

Dunne, Anthony. 2006. "Hertzian Tales". MIT Press, Cambridge, MA, USA.

Ermi, Laura; Mäyrä, Frans. 2005. "Fundamental Components of the Gameplay Experience: Analysing Immersion". In Proceedings of the 2005 DiGRA International Conference: Changing Views: Worlds in Play, vol. 3. Vancouver, CA.

Frankel, Lois; Racine, Martin. 2010. "The Complex Field of Research: for Design, through Design, and about Design". Retrieved on October 22, 2018, from <http://www.drs2010.umontreal.ca/data/PDF/043.pdf>

Garret, Jesse James. 2002. "The Elements of User Experience: User-Centered Design for the Web and Beyond". New Riders, San Francisco, CA, USA.

Grinblat, Jason. 2017. "Designing for Modularity". In T. Short and T. Adams (Eds.) *Procedural Generation in Game Design*. A K Peters/CRC Press, New York, NY, USA.

Jørgensen, Kristine. 2013. "Gameworld Interfaces". MIT Press, Cambridge, MA, USA.

Juul, Jesper. 2013. "The Art of Failure: An Essay on the Pain of Playing Video Games". MIT Press, Cambridge, MA, USA.

Khaled, Rilla; Lessard, Jonathan; Barr, Pippin. 2018. "Documenting Trajectories in Design Space: a Methodology for Applied Game Design Research". In *Proceedings of FDG 2018: 13th International Conference on the Foundations of Digital Games*. Malmö, SE.

Lazzaro, Nicole. 2004. "Why We Play Games: Four Keys to More Emotion in Player Experiences". In *Proceedings of the Game Developers' Conference*, San Jose, California. USA. Retrieved on October 11, 2018, from http://twvideo01.ubm-us.net/o1/vault/gdc04/slides/why_we_play_games.pdf

Lessard, Jonathan; Arsenault, Dominic. 2016. "The Character as a Subjective Interface". *Lablablab.net*. Retrieved on October 22, 2018 from https://lablablab.net/papers/SubjectiveInterface_ICIDS2016.pdf

Lessard, Jonathan; Brunelle-Leclerc, Etienne; Gottschalk, Timothy; Jetté-Léger, Marc-Antoine; Prouveur, Odile; Tan, Christopher. 2017. "Striving for author-friendly procedural dialogue generation". In *Proceedings of tFDG 2017: 12th International Conference on the Foundations of Digital Games*. Hyannis, MA, USA.

Matulef, Jeffrey. 2016. "Event[0] is 2001 meets Firewatch, due this September". *Eurogamer.net*. Retrieved on november 23, 2018 from <https://www.eurogamer.net/articles/2016-07-13-event-0-is-2001-meets-firewatch-due-this-septembe>

Nakamura, Jeanne; Csikszentmihalyi, Mihaly. 2002. "The concept of flow". In C. R. Snyder and S. J. Lopez (Eds.), *Handbook of positive psychology*. Oxford University Press, New York, NY, US.

Novak, Jeannie; Saunders, Kevin. 2012. "Game Development Essentials: Game Interface Design". Delmar Publishers Inc., Clifton Park, NY, USA.

Quinten, Niels; Maillet, Steven; Conix, Karin. 2017. "Gaps of Uncertainty: a Case for Experimentation in Serious Game Design Frameworks". In P. Lankoski, J. Holopainen (Eds.) *Game Design Research: An Introduction to Theory and Practice*. ETC Press, Pittsburgh, USA.

Ryan, James; Seither, Ethan; Mateas, Michael; Wardrip-Fruin, Noah. 2016. "Expressionist: An Authoring Tool for In-Game Text Generation". In *Proceedings of Interactive Storytelling: 9th International Conference on Interactive Digital Storytelling*. Los Angeles, CA, USA

Salen, Katie; Zimmerman, Eric. 2004. "Rules of Play". MIT Press, Cambridge, MA, USA.

Samuel, Ben; Ryan, James; J. Summerville, Adam; Mateas, Michael; Wardrip-Fruin, Noah. 2016. "Bad News: An Experiment in Computationally Assisted Performance". In In Proceedings of Interactive Storytelling: 9th International Conference on Interactive Digital Storytelling. Los Angeles, CA, USA

Schell, Jesse. 2015(2008). "The Art of Game Design: A Book of Lenses". CRC Press, Boca Raton, FL, USA.

Sicart, Miguel; Wilson, Doulgas. 2010. "Now It's Personal: On Abusive Game Design". In FuturePlay 2010. Vancouver, Canada.

Swink, Steve. 2008. "Game Feel". CRC Press, Boca Raton, FL, USA.

Wardrip-Fruin, Noah. 2009. "Expressive Processing". MIT Press, Cambridge, MA, USA.

WORKS CITED

Bad News, designed and developed by James Ryan, Adam J. Summerville and Ben Samuels, 2015.

Dark Souls, developed by From Software, published by Namco Bandai Games, 2011.

Dear Esther, designed, developed and published by The Chinese Room, 2012.

Event[0], developed and published by Ocelot Society, 2016.

Five Nights at Freddy's, designed and developed by Scott Cawthon, 2014.

Gone Home, designed by Steve Gaynor, developed and published by The Fullbright Company, 2013.

Hamurabi, designed and developed by Doug Dymment, 1968.

Pacman, designed by Tory Iwatani, developed by Namco, published by Namco and Midway, 1980.

Paper's, Please, designed by Lucas Pope, developed and published by 3909 LLC, 2013.

Shrouded Isle, designed, developed and published by Kitfox Games, 2017.

Sid Meier's Civilization V, developed by Firaxis and published by 2K Games, 2010.

Sim City, designed by Will Wright, developed and published by Maxis, 1989.

The Sims, designed by Will Wright, developed and published by Maxis, published by 2000.

The Stanley Parable, designed by Davey Wreden and William Pugh, developed and published by Galactic Cafe, 2013.

The Uncomfortable, designed by Katerina Kamprani, available online at <https://www.theuncomfortable.com/>, 2017.

The Walking Dead. Developed and published by Telltale Games, 2012.

Tracery, designed and developed by Kate Compton, 2014.