# Values at Play: Design Tradeoffs in Socially-Oriented Game Design

**Mary Flanagan**
Tiltfactor Lab, Film & Media
Hunter College
New York, NY, USA
mary@maryflanagan.com

**Daniel C. Howe**
Media Research Lab
New York University
New York, NY, USA
dhowe@mrl.nyu.edu

**Helen Nissenbaum**
Dept. of Culture & Communication
New York University
New York, NY, USA
helen.nissenbaum@nyu.edu

**Abstract**

Significant work in the CHI community has focused on designing systems that support human values. Designers and engineers have also become increasingly aware of ways in which the artifacts they create can embody political, social, and ethical values. Despite such an awareness, there has been little work towards producing practical methodologies that systematically incorporate values into the design process. Many designers struggle to find a balance between their own values, those of users and other stakeholders, and those of the surrounding culture. In this paper, we present the RAPUNSEL project as a case study of game design in a values-rich context and describe our efforts toward navigating the complexities this entails. Additionally, we present initial steps toward the development of a systematic methodology for discovery, analysis, and integration of values in technology design in the hope that others may both benefit from and build upon this work.

**Categories & Subject Descriptors:** H.5.2 [Information Interfaces and Presentation]: User Interfaces—prototyping, evaluation/methodology, human factors

**General Terms:** Design, Human Factors

**Keywords:** Gender and Computing, Programming Pedagogy, Social Issues, Values

## INTRODUCTION

Significant work in the CHI community has focused on designing systems that support human values. This work sets forth values as a design aspiration, exhorting designers and producers to include values in the set of criteria by which the excellence of technologies is judged. Research has focused, for example, on the value of privacy [1,2], community [13], freedom from bias [23], universal usability [46], autonomy [48], informed consent [19], and trust [16,22,38]. Designers and engineers have also become increasingly aware of ways in which the artifacts they create can embody political, social, and ethical values. It is one thing to subscribe, generally, to these ideas, even to make a pragmatic commitment to them, but putting them into practice in the design of technical systems such as software games is not straightforward. Experienced designers will recall the not too distant past when interface and usability were overlooked features of software system design. While these and other aspects of design have now entered the mainstream of research and practice, we are still at the shaky beginnings of thinking systematically about design and values. Even those designers who support the principle of integrating values into systems are likely to have trouble applying standard design methodologies, honed for the purpose of meeting functional requirements, to the unfamiliar turf of values. What seems absent from much of the literature on values and technology design—specifically, computer software design—is a set of explicit pointers to guide design whose successful endpoint is the reliable capacity to embed values in software systems. This is a crucial missing piece that could help those motivated by the principle to move beyond mere exhortation.

Our team recognizes that there are very different approaches to defining 'values'; most are culturally or socio-economically specific. The modest goal of this paper is not to assert which values should be of importance, but instead to demonstrate how to discover relevant values for a particular project. Not unexpectedly, many designers struggle to find a balance between their own values, those of users and other stakeholders, and those of the surrounding culture. In this paper, we present the RAPUNSEL project as a prime example and case study of design in a values-rich context and describe our efforts toward navigating the complexity this entails.

The aim of this paper is to help fill in this missing element by sketching a methodological framework for the consideration of values during the process of design. In RAPUNSEL, a three-year, NSF-funded project, a team of computer scientists, interaction designers, and social psychologists were tasked with the creation of a networked game environment to teach programming to middle-school girls. Although it is a large project with multiple interlinked



*CHI 2005,* April 2–7, 2005, Portland, Oregon, USA.

components (e.g. pedagogy, interface, graphics, networking, etc.), challenging questions about values emerged in several key phases. It was therefore essential to the project's success to both iteratively address questions concerning values and systematically implement our answers in the design. The project is still in progress—the second year of a three-year project; therefore, the system is being built alongside this methodological outline. Here, we focus on our design process and values investigation.

Drawing on a number of existing approaches and analytic frameworks,[1] we discuss the set of values that were considered over the project's lifecycle. Additionally we present specific examples of values conflicts we encountered and their subsequent resolutions. Our approach is not intended to replace well-established design methodologies (such as user-centered design, participatory design, or the iterative design methods specific to game design noted by [12,51], but rather to demonstrate with concrete examples the way in which attention to values in the design process can inform the stages of many existing design processes. In this way, we hope that software design projects, and in particular, social software and game development, might adopt this approach as a hybrid methodology. Prior work influencing this framework includes several other approaches to integrating values in design, including Value Sensitive Design, Participatory Design, Reflective Practice, and Critical Technical Practice.

The significant contributions of this paper are three-fold. First, we present a case study of an interdisciplinary design process in a particularly values-rich context. Second, we present an overview of frameworks and methods which address such contexts, iterating the features of each that we employed in the project. Third, we present a set of specific steps that we employed to discover, analyze, and integrate values within the system. While many designers will recognize the familiar clashes between designer and client, social values and stereotypes, as part of their daily practice, there are few systematic ways (such as, for example, in usability heuristics) in which values can be discovered, interrogated, and integrated into software design projects. It is our hope that such work will help to shed light on the benefits and challenges of employing a values-oriented approach across a variety of design contexts, especially those with socially-oriented and/or educational components.

## THE RAPUNSEL PROJECT: AN OVERVIEW

RAPUNSEL (http://www.rapunsel.org) is an NSF-funded research project in which collaborators from Hunter College, New York University, and the University of Illinois are designing and building a game environment for teaching middle-school girls, particularly those from disadvantaged home environments, to program computers.[2] Concerned by data that shows a radical decline of interest in math, science, and computing in female adolescents [6,8,9, 11,30,49], but noting strong evidence suggesting that networked software environments and social learning appealed to girls, project developers sought to correct this imbalance by developing socially-oriented environments for learning these subjects. Games, especially online games, are a significant pastime for the target audience: sixty-five percent of Internet game users are female [45] and 12.4 million US teenagers use instant messaging regularly. Social software is especially popular with teens [10,26].

Informed by this research, we developed the RAPUNSEL game, an engaging dance-driven set of activities which embeds the pedagogical goal of teaching programming within a group space to address gender imbalances in technology education. In the game, programming is an essential skill for navigation, interaction, and play.

The RAPUNSEL game is set in a dance-driven environment populated primarily by two groups of creatures: Peeps and Gobblers. Although both like to dance, the two groups are rivals; Gobblers are aggressive, and regularly kidnap little Peeps to use in their own dance routines. The Gobblers live down in the Underworld, while Peeps live up in the trees. Players control Peeps by programming dance sequences. They are introduced to the Java programming language through scaffolded, guided exercises where they first teach characters to do specific moves, then how to dance in a complex, synchronized fashion. Players first work with one Peep with simple move methods, then learn more complicated programming concepts such as loops and conditionals, using these moves to ultimately prepare for dance competitions with Gobblers and with other players. Players may choose to play along with this narrative in a competitive mode of interaction, working to be the top online choreographer/coder, or players may engage in an 'exploratory' mode of play, choosing to collect and swap code snippets, decorate their homes or make music to use in their own dance sequences.

Early on, the goal of the project migrated from simply teaching programming to doing so in a way that encourages our target demographic to engage with the computer in creative, social, and novel ways, thereby empowering users in the larger contexts of math, science, and technology [28]. We have continually examined alternative reward structures for the game environment based upon increased social presence in homes of creating a "common ground" for players to interact and learn; by increasing social presence, this approach may also improve enjoyment for the activity [14].

[1] We take as a starting point the analytic framework proposed by Nissenbaum et al. [37,23] known as Values in Design, and augment it with insights from related methodologies.

[2] The Principal Investigators of the RAPUNSEL project are Ken Perlin (NYU), Mary Flanagan (Hunter College), and Andrea Hollingshead (UIUC).

It is our hope that this approach, in combination with appropriately scaffolded programming fundamentals, will best result in reaching the goals of the project.


**Figure 1. RAPUNSEL prototype: Code Editing.**

## SIGNIFICANT VALUES DIMENSIONS IN THE PROJECT

Since the project's beginnings, we believed that RAPUNSEL would contain significant values dimensions. Our reasons for such an assessment follow.

1. Because a core project goal was to alter status-quo perceptions and biases concerning gender and enact change in areas of access and empowerment for girls, the project context was politically charged [3,33].
2. The project involved an unusually large number of stakeholders from highly diverse backgrounds. Beyond our primary target audience, we were compelled to seek and integrate feedback from teachers, parents, and industry representatives; all interested parties with diverse political, social, and ethical views.
3. Whenever games are created, difficult questions arise concerning character representation, including gender and race (oversexed characters), social and hierarchical rewards (advanced players acquire wealth and power), interaction styles (killing vs. protecting), representation, and a host of related values questions requiring analysis.
4. The project is multi-disciplinary in nature, involving educators, social scientists, computer scientists, artists, designers, and students from a range of backgrounds. This collaboration proved well-suited to a values-based approach as values could be continuously queried from different points of view.

## EXISTING METHODOLOGIES FOR INTEGRATING VALUES

We drew from several existing frameworks to inform the values-oriented aspects of our methodology.

### Values-Oriented Approaches

Values in design is an emerging umbrella term for a related group of interdisciplinary approaches to systematically identifying and accounting for values in technology design. This general approach builds upon the thesis that technical systems often come to embody values. This is a complex claim put forward in various forms by a number of prominent philosophers and social scientists [7,27,31,35,37,50; L. Lessig, D. Mackenzie, M. Akrich], with about as many variations on what it means.

Overall, values in design commits to the pragmatic goal of designing systems so that they embody values to which designers, users, other stakeholders, and the surrounding society are committed. In a liberal democracy, these might include liberty, autonomy, justice, freedom, security, and privacy (merely a few examples of what may be an indefinitely long list). Values in design poses a challenge to those involved in system design to adopt values as one among a set of criteria according to which system quality is judged. But *how* to meet this challenge, how to design *for values* amidst complex factors including pre-existing bias, has remained an open question whose ideal answer would contain, among other things, a clearly and fully articulated methodology.

Value-Sensitive Design attempts to further values in design research by proposing a methodological approach for the study of values in technology [18,20,21,22]. In designing for RAPUNSEL, we build upon a three-part framework proposed by these researchers, incorporating elements from three primary areas: the conceptual, the empirical, and the technical**.** In the design context, these elements are thoroughly intertwined.

The *conceptual* aspect of the VSD methodology considers what values are at stake in a project and gives rise to an initial set of appropriately defined values. Designers operating in the US, for example, are likely to be influenced by values embodied in core political documents like the Constitution. In RAPUNSEL, we began by asking what values we hoped to support in order to be sure these were integrated into the design of our research questions, our initial prototypes, and eventually, our implementation. Our specific list focused on values of autonomy, equality, access, empowerment, democracy, and authorship. A more complete list of values emerged as the project progressed that included creative expression, subversion, system transparency, group success, community, collaboration, diversity, and fair representation.

The *empirical* component of our work helped us to ascertain the value preferences of those affected by the system under study. The clearest need for this occurred when trying to resolve value conflicts in the context of design choices. Relevant data was often available from previous and concurrent research; empirical investigations for RAPUNSEL ranged from the collection of relevant documents, to paper and online surveys, to observations and probed interviews, to quantitative measurements collected from within the software. The team also examined game systems that appealed to our target demographic, previous

tools for teaching programming, the various programming languages created for children.

Defining *technical* demands is a complex endeavor, as systems will generally have some number of purposes that are not values-oriented in nature. A technology can be best suited for certain activities and can thus support certain values—that is to say, certain technological properties can help or hinder particular values [21]. For example, online blogging systems may not be created to prioritize the value of privacy as would a database system in which medical records are maintained. Fingerprint readers are a technology which imply strict control of privacy and identity and would not appropriate for use as, say, a requirement to access a general search engine such as Google. The technical requirements posed in RAPUNSEL include a need to support chat and a need to protect user identity to support the other project values of community, group success, and collaboration.

While values in design investigations, and the VSD methodology in particular, provided a foundation for our approach to managing values in RAPUNSEL, much of the work in this area has focused on analyzing systems already built, rather than building systems from the ground up. This prompted our team to look to methods more directly engaged with the dynamic nature of the design process.

### Reflective Practice

The philosophy of reflective practice focuses on how practical examples and methods can complement the philosophical and theoretical exploration of a research theme. The description of this 'reflexive conversation' is perhaps best articulated by D. Schön, though it has become a central focus in the work of many other researchers across disciplines. Schön encourages practitioners to become 'reflective practitioners', that is, to participate fully in a dialogue with one's actions by thoroughly understanding the design problem and noting discrepancies between one's beliefs and one's actions. For example, in RAPUNSEL, the question "how to teach programming to middle school girls" was reframed as, "how to create a compelling environment in which programming is a central element". This reframing stresses the playability and sociability of the environment along with the central concern of programming. In this way, reframing affects the larger 'web' of the project [44]. Other reflective frameworks, such as 'critical technical practice' are advanced primarily by computer science practitioners in artificial intelligence such as [2], S. Penny, and M. Mateas.

### Participatory Design

Participatory Design (PD) is an approach to the design, creation, and study of systems in which the users become primary participants in the design process [13,25,34]. In a PD approach, technologies are studied as holistic systems, including the people, practices, and organizations that may interact with them. PD is often employed as a means for allowing workers to have input in the technologies that affect their work—it focuses on designing *with* people, not merely for people. Among other concrete practices, prototyping is essential to PD, and similarly is essential to our own process in RAPUNSEL [5,36]. Other PD principles useful to our team include a focus on contexts-of-use in collaboration with our 'design partners'. In RAPUNSEL, regular meetings with groups of co-designers help us to understand what themes and subjects girls prefer, when and how the software would be used, and the technical limitations of these situations. Similarly, the YP programming principles of working in short iterations, continuous integration, testing, and collective code ownership facilitates the integration of numerous participants' feedback in the design process; this approach is helpful when working in 'politically tempestuous domains' like RAPUNSEL or the UrbanSim project at UW [17], in which the politically charged domain of urban planning is supported through the use of these programming methodologies. Toward this end, we also applied specific aspects of 'Agile Programming', a related methodology useful in rapidly integrating participation and input from diverse stakeholders [4].

### Game Design

While not fully engaged with values in the design process, methods from game design support similar approaches to PD and Reflective Practice. Crawford's description of the game design sequence examines game making as an artistic and technical process, with prototyping and playtesting used to support a particular game's clearly defined goals [12]. In addition to a postmortem process, reflection on the design occurs in Crawford's model from the research and design phase to the playtesting phase. This is in line with the use of iterative cycles between playtesting prototypes [51] and research with prototypes and game preferences [32]. Playtesting games can be a time to discover and verify values in a particular game design. Games replicate the value systems of the culture or community in which they are created and played [43].

## A METHODOLOGICAL APPROACH TO ITERATIVELY DISCOVER, ANALYZE, AND INTEGRATE VALUES

The combination of methods employed in RAPUNSEL provided a theoretically grounded approach that allowed the team to consider human values in a rigorous and iterative manner throughout the design cycle.

### Values Discovery

The first step in our analysis of relevant values we term 'discovery'. To be committed to better design practices, we need to adopt systematic steps to query the values motivating them. The key heuristic we suggest for answering the question, "what values?" is to reflect on the sources of values both generally and in relation to any given technical system or device. Sources of values are entities, including individuals, institutions, societies, and cultures that suggest relevant values or place values-oriented demands on creators.

While at the start we believed that the majority of important values would be based in our initial project plans as explicitly stated values, it became clear that new values continued to emerge throughout the project's development. Consequently, examining these emergent values became part of our iteration cycle. Here, we enumerate sources of values which we have found to be of use. While this exact set may not be either sufficient or exhaustive across contexts, we believe that it provides a useful set of starting points that can be modified and augmented as needed. These sources include the explicitly stated project goals, the hypotheses generated by the team to achieve those goals, the values expressed in prior empirical work, including related technical systems, values present in the design environment (academia, commercial, activist, etc.) and values held by individual members of the design team.

*1. Project Goals*

The first activity of discovery involves answering the question: what values are implicated by, inspire, or inform a given design project? The ideal starting point for discovery is with the explicit project goals as described within documents such as project proposals, or client meetings and presentations. Values found here tend to be 'higher-order' values, that is, values we perceive as ends in themselves. In the case of RAPUNSEL, these included autonomy, equity, access, creativity, diversity, empowerment and authorship. These goals were formulated "to address gender inequities and address the needs of a sector overlooked by the software industry." Values expressed in such a project description—in this case, gender equity—are part of the definition of the project. Yet such an explicitly expressed value can also lead to other, more concrete, values. In this example, our belief that fluency with IT in a Western, technologically literate society holds personal, cultural, and financial currency and thus promotes the larger values of equity and autonomy. Concretely conceived values may be more salient in certain communities (such as technical skills) in the pursuit of more abstractly conceived values, such as equity.

Our team feels it necessary to periodically revisit the initial project goals to verify that they are indeed supported in our process. On several occasions we realized that we had strayed from one or more of these basic project goals as we focused on the navigation of particularly thorny issues; an example here involved the recurring question of how to implement computer programming as a central and organic component of the game mechanic. As we grappled with this challenge, that is, how to not only include programming in the game play, but to make it an effective and enjoyable focus, we found ourselves unknowingly sacrificing other basic goals in the process. Zimmerman notes that in the development of games, designers must identify play values, the abstract principles that the game design will embody [51]. In the RAPUNSEL project, however, the social values and the play values had to be continuously aligned.

*2. Hypotheses*

After clarifying high-level project goals, we developed a set of hypotheses that might realize these goals. These hypotheses also proved to embed certain values, though these were generally more of an instrumental nature; that is to say, values held primarily in support of other values as opposed to ends in themselves. For example, an important project hypothesis was that successfully learning programming would emerge from immersion in a socially-oriented game environment in which programming was a central activity in the interaction. Appealing to girls' interest in online social interaction might help successfully teach programming concepts, thereby helping to achieve higher-level goals [29,39,40,47]. Thus, in this case, several instrumental values (collaboration, engagement, social interaction) supported the higher order project value of learning programming.

*3. Prior Work*

Prior work in the form of published papers, books, empirically collected data, and existing technical systems continues to prove to be a fruitful source of values. While much such work originated in Computer Science, we additionally searched across a broad range of disciplines that included Cognitive and Developmental Psychology, Game Studies, Linguistics, Feminism, and Cultural Studies, to name a few. Additionally we researched existing technical systems and components including scripting languages, software editors, interpreters, renderers, game engines, as well as popular game titles that interested our target audience such as *Grand Theft Auto: Vice City* and *The Sims*. Papers by educators and scientists in the area of computer science education, along with existing software environments created by both researchers and commercial parties were studied. The success of the LOGO programming language, for example, led to the inclusion of values such as transparency [39] and reinforced values of empowerment and authorship. Similarly, prior work by game researchers that focused on values issues led to the inclusion of additional values such as creative expression, group success, diversity and fair representation [15,32].

*4. Designer Values*

While not often analyzed, the beliefs of designers themselves often have significant effects upon the values embodied in a particular system and thus are an essential component of the discovery process. The incorporation of 'creativity' or self-expression within RAPUNSEL is an example of such a designer-introduced value; one that became apparent to the team only after it emerged in prototypes exploring other issues. Once 'discovered' and discussed, it became clear that this value was of import to several members of the team and was then included in our list of explicit values. While this proved possible in some cases, there were others, as described later, where designer values proved to conflict with core project goals.

*5. User Values*

User testing, in the broad sense, has become an increasingly important aspect of systems design in recent years and designers have continually struggled to more accurately assess the value preferences of their users. While it is always important to assess exactly what dimensions people most care about in regards to a technology, it is particularly so when trying to resolve value conflicts in design. For example, in deciding what degree of anonymity to provide users in, say, an email system, it might be important to ascertain how each might rank the capacity to speak freely without fear of retribution, in comparison to the ability to hold speakers accountable for what they say.

This process of regularly prototyping values-based design decisions with users proved an essential component of our process. Within this category we include a variety of prototyping, ranging from focus-group style and one-on-on sessions with our design partners, to web-based survey collections, paper-prototyping, digital mock-ups, and more traditional testing modules implemented in software. Such prototyping sessions proved to be a useful source of values, with values related to representation, self-expression and authorship emerging here. Values that emerged from users included the value of subversion (playing games such as *The Sims* "wrongly" was a significant pastime for our design partners), which led us to recognize this behavior as part of a larger, emergent value related to autonomy. Other values discovered here included users' desire to build and dress-up characters (authorship and creative expression), their desire to manipulate characters in the game to engage in deeper relationships with each other via flirting, dancing, and other social behaviors (community, autonomy, authorship, and collaboration).

*6. Stakeholders*

Parents, educators, funders and peers (software and design professionals) are all stakeholders in an educational software development process. As such, game designers, educators, philosophers, artists, parents, and industry experts were all invited to attend RAPUNSEL meetings, meet in special think-tank sessions, and/or offer commentary electronically. Such interaction provided a steady stream of new input where new values-dimensions emerged throughout the project. For example, both the funding organization (the NSF), and various colleagues from the industrial sector expressed views related to the economic value of gender diversity in technological fields.

Finally, it is important to re-emphasize the iterative nature of this discovery process. Values not only appear throughout the process but can also change in importance and even type. For example, what one believes to be an instrumental value may change into a higher-order value when its importance is stressed by users, stakeholders and designers. To appropriately handle such a dynamic, it is important that the team remain flexible and aware of the fact that conclusions within any single sphere (for example, user-feedback) are never final, but open to revision as suggested by input from others sources.

## IDENTIFYING VALUES-BASED CONFLICTS

An important element in our process was the consideration of *values-based conflicts* in the context of particular design choices. Engineering is rife with such conflicts – should we emphasize safety or cost, transparency or privacy, and many more. The case of software design, as practitioners are well aware, is not significantly different in this respect. In the terms of our framework, conflicts occur when designers find themselves unable to implement all values to which they have committed—or *discovered*. Those familiar with the enormous literature in philosophy, law and politics will understand why we cannot presume to offer a simple solution in such situations, though we are aware of a variety of systematic approaches to resolving conflict that have been proposed [42]. Our experience with RAPUNSEL, and presumably this is true in other practical situations where conflict arises, is that certain strategies prove particularly useful. One such strategy involved the clarification of values and design elements which allowed us, in some cases, to dissolve conflicts; that is, to show that one could settle on a decision that avoided choosing one value over another. In other cases, we resolved conflicts by agreeing that some commitments outweighed others and hence the lesser commitments would be traded-off in design decisions.

At various times in the project we found that certain project goals were in conflict with specific values of users. For example, while complex social situations provide a more motivating learning environment for the girls in our target demographic, social engagement tended to draw attention away from goal-oriented activities like programming. Once this tension was recognized, it was added to our list of conflicts, each of which was assessed in subsequent design iterations—much in the way that regression testing was employed in validating our continuous technical progress.

Throughout the RAPUNSEL design process, the team noted that particular values emerged outside those of the original project goals—sometimes for the betterment of the project, sometimes not. For example, when the question arose, early on, about how to devise a reward structure for the game environment, designers first reasoned that a care-giving or nurturing structure would work best due to the popularity of such games with the target audience. However, further research and prototyping showed that this initial conception was incorrect. Rather than promote the values of cooperation and collaboration, this original game design fostered a quite competitive style of care-giving (such as found in the *NeoPets* game). The initial design thus led to a values conflict. Later design iterations established a cooperative reward structure that encouraged sharing of elements and ideas between players—goals better matched to empirical findings on girls' science and mathematics learning preferences [3,6].

Another example of conflicting values emerged in consideration of the game's code interface. Design partners stated that they did not appreciate the need to type their own code (a feature we had associated with empowerment), but rather preferred to directly manipulate the environment via mouse input. This sentiment conflicted with an initial project hypothesis gleaned from informal interviews with colleagues and students who had learned to program in 'drag-and-drop' environments (i.e., Director, Flash, Max/MSP, etc.), that 'GUI-programming' placed unnecessary distance between the user and the system that frustrated learning. Thus, our belief that the achievement of programming skill and expressivity was likely to require the actual typing of code was in direct conflict with user values of ease-of-use and efficiency, not to mention the ubiquity of direct-manipulation interfaces in games and other software. Our solution to this conflict was to build a scaffolded interface system that began with direct manipulation, transitioned to context-sensitive pull-down menus, and finally required typing in a 'smart', assistive code editor. The success of this interface was realized by allowing additional dimensions of freedom with each successive transition, so that typing became available when users required higher degrees of expressiveness than the simpler interface afforded. Thus, the actual typing of code became associated with reward rather than unnecessary effort.

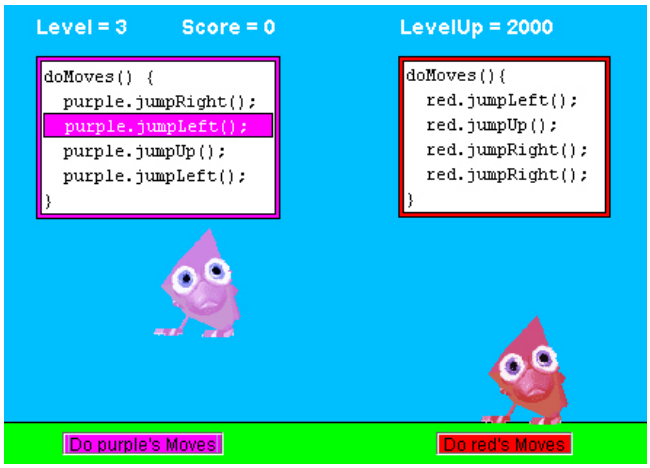


**Figure 2. RAPUNSEL prototype: Code Sequencing.**

## IMPLEMENTATION AND PROTOTYPES

### The Prototyping Process

The continual development of small, targeted prototypes as discussed by [12,51] allows game developers to quickly test solutions to a variety of issues, including many with important values dimensions, before any full model of the system exists. In RAPUNSEL, we began by developing small prototypes of game-play scenarios and character behaviors, temporarily ignoring difficult issues such as the scaffolded learning system and complex tools like the 'intelligent' script editor. The goal of creating 'throw-away' prototypes was to quickly discover flaws in the design of major system features. This process allowed us to solicit immediate feedback, even in the first weeks, via paper prototyping, which drove software implementation and helped to mitigate high-risk areas. Paper prototypes, as documented by [36,41] often prove highly effective in quickly determining the applicability of conceptually oriented features. In RAPUNSEL, features which required early and continual feedback included social and cooperative interaction design and the use of alternate game goals to motivate various learning styles, both of which emerged from initial design hypotheses and manifested important instrumental values.

As significant research suggests [5,17,46], working in tight, iterative cycles proves most effective in facilitating the incorporation of feedback from the full range of stakeholders. Throughout the project, we iterated new designs after meeting with design partners, educators, and industry advisors and verified that each could be mapped back into both the technical and values-oriented frameworks under development. The team settled into a pattern in which rough technical sketches could be created quickly enough to reflect emerging data while not incurring extensive time/effort in implementation. This was particularly important as the team was small and most members worked across typical areas of specialization. As a consequence of such resource constraints, it was important that prototypes were differentiated into two types: those with a short life span which were not likely to be modified (and thus were created as fast as possible without consideration of reuse or modularity), and those that might be integrated into the larger project framework. Even in this latter case it was essential to both maintain a rapid prototyping pace to incorporate new ideas while effectively controlling resources. Potential framework components could be developed quickly based upon functional requirements and then refactored for architectural coherence only when their inclusion in the technical framework became necessary.

A final point regarding prototypes concerns the importance of formulating specific questions that each prototype is designed to address. To facilitate this goal in a systematic way, each RAPUNSEL prototype was grouped in one of the following categories: Game-play, Technical, Aesthetic, Character, Simulation, and Behavior. Once assigned a category, the prototype's primary question was often readily apparent. This was essential to maintaining design momentum and largely avoided problems pointed out by software engineers [24] who note that prototypes created with unclear or conflated goals can serve to frustrate a design process by providing answers to the 'wrong' questions. As Glass argues, poor design often yields prototypes that attempt to answer questions in the general category of 'can we do this?'—the answer to which is almost invariably yes [24]. In our experience, successful prototypes rarely tested what was possible (with the exception of a few technical issues) but rather what was viable and most closely reflected core projects goals and values. If a prototype's question was not clearly framed by

such core values, design choices that followed were unlikely to manifest them.

An example of the prototype implementation process in RAPUNSEL again involved our reward system, and, in particular, the design of the code sharing system within the game. After clarifying the core project value of collaboration and considering various implementation strategies, we decided to create a test system in which players could compose, accumulate, and transport the code segments they create with them through the various stages of the game in virtual 'backpacks'—similar to how one might gather weapons or armor in traditional conflict-oriented games. In encounters between players, one player may then query the other to learn what pieces of code they are carrying, and the other might then agree to share some or all of the code segments in their backpack. Each time a player's code is used by another player in the game, the originator receives points, thus encouraging not only sharing, but authorship and creativity as well.

On a technical level, the implementation of sharing and collaboration in the manner described grew complicated as we considered how players might save and transport pieces of code across various stages of the game. Additionally, it was important that each code segment be created and saved at the same level of granularity. Requiring the code to conform to a high-level unit like a class or interface, however, seemed likely to be too difficult for new players to grasp, and could discourage sharing among less experienced players. At the same time, to satisfy usability constraints, it was essential that players could easily keep track of, share, and use the code which they had previously saved.

## VERIFICATION

As discussed above, testing with prototypes proved an important tool in verifying that design decisions adequately handled the complexities of values-oriented tradeoffs. In such testing, it was necessary to determine not only that a particular value was successfully implemented in a technical component but also that such an implementation did not detract from prior decisions, either functional, interactive, or conceptual in nature. Concurrently it was essential that playability, entertainment-value, and pedagogical components were not weakened by new decisions. Balancing these goals with often competing objectives proved a central challenge in RAPUNSEL—one that might have been impossible to meet had we taken a less systematic approach to handling design tradeoffs or not consistently prototyped our resolutions with specific value dimensions in mind.

An example of verification in RAPUNSEL involved the code-sharing aspect of the game discussed previously. During game play, players spend significant time saving code to use and share with others. To verify this aspect of the game, it was important to closely examine players' motivation for such saving and sharing. Although the system clearly provided the capability for code sharing, it was essential that we verify players' motivations to do so.

Through several prototype iterations, we resolved these issues by integrating code-sharing directly into the smart code editor, and combined code saving and sharing with an important pedagogical goal we hoped to teach early in the game; namely that of teaching encapsulation, a key programming concept. The design rewarded players with the accumulation of knowledge, as represented by code segments, rather than with material items, like weapons, clothing, or money, and thus reinforced other core project values: specifically sustainability and non-violence. The players' motivation to share was clear, as it was the primary way to accumulate points and advance through the game.

## CONCLUSION

The process we have detailed in this paper is as follows:

1) Values Discovery: Create a working list of relevant values from value sources including Project Goals, Hypotheses, Prior Work, Designer Values, User Values, and Other Stakeholder Values.
2) Identifying Values-Based Conflicts: Check functional components for values conflicts.
3) Implementation and Prototyping:
   - Work through values conflicts generated in specific functional components.
   - Categorize the prototype: In our case, we used a) game-play; b) technical; c) aesthetic; d) character; e) simulation; f) behavior.
   - Formulate a specific question for each prototype.
   - Work in iterative cycles with reflective review, involving participants, such as design partners, to incorporate ongoing feedback.
4) Values Verification: From the list of initial and emergent project values, verify that desired project values are embedded in the project and others are not.

We see the RAPUNSEL project as 'activist' because our high-level goals are not only to teach programming, but to do so in manner that promotes significant social change in equity, empowerment, and access to technology. Laurel notes that "values are everywhere, embedded in every aspect of our culture and lurking in the very nature of our media and our technologies" [32, p. 62). The long list of values discovered in RAPUNSEL demonstrates how charged the development of a socially-oriented educational game can be. But our work goes further than recognizing and listing potential values in a design project. It seeks to articulate specific steps for designers to follow when grappling with values while facing real issues and challenges of large-scale software design. This initial methodology for design in value-rich contexts includes a

toolset which we hope will be useful to the designers of games and educational software. While the primary purpose for developing the methodology was to facilitate the integration of values into RAPUNSEL, we hope that this work will also contribute to systematic approaches for others designing in value-rich contexts.

## ACKNOWLEDGEMENTS

There are far too many scholars, designers, technologists, teachers, artists, and children involved in both the RAPUNSEL and Values in Design projects to thank them individually; we must extend gratitude to those who are grappling with values issues, and encourage efforts for larger social change through both theory and practice. We would particularly like to thank each and every member of the RAPUNSEL team for their dedicated work on the project, especially the Co-PIs Perlin and Hollingshead. RAPUNSEL is supported by NSF's Program Research on Gender in Science and Engineering, HRD0332898. Values in Design research is supported by the Ford Foundation.

## REFERENCES

[1] Ackerman, M.S., & Cranor, L. Privacy critics: UI components to safeguard users' privacy. In *Ex. Ab. CHI 1999*, ACM Press (1999), 258-259.

[2] Agre, P.E. Introduction. In P.E. Agre & M. Rotenberg (eds.), *Technology and Privacy: The New Landscape*. MIT Press, Cambridge, MA, USA (1997), 1-28.

[3] American Assn of Univ, Woman (AAUW).*Tech-Savvy: Educating Girls in the New Computer Age*. (April 2000). Available Online: <http://www.aauw.org/2000/techsavvy.html>

[4] Beck, K. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, Boston, MA, USA (1999).

[5] Bødker, S., & Grønbæk, K. Design in action: From prototyping by demonstration to cooperative prototyping." In J. Greenbaum & M. Kyng (eds.), *Design at Work: Cooperative Design of Computer Systems*. Erlbaum, Hillsdale, NJ, USA (1991).

[6] Brunner, C. Opening technology to girls: The approach computer-using teachers take may make the difference. *Electronic Learning,* 16:4 (February 1997), 55.

[7] Brey, P. (2000) Disclosive Computer Ethics, *Computers and Society*, Vol. 30: 4 (2000) 10-16.

[8] Catsambis, S. The path to math: Gender and racial-ethnic differences in mathematics participation from middle to high school. In *Sociology of Education* 67 (1994), 199-215.

[9] Chaika, M. Ethical Considerations in gender-oriented entertainment technology. In *Crossroads of the ACM*. (November 1995). Available Online: <http://www.acm.org/crossroads/ xrds2-2/gender.html>

[10] Chmielewski, D.C. Kids turning to instant messaging. Knight Ridder, (25 Feb 2004). Available Online: <http://www.azcentral.com/families/articles/0225famin stantmessage.html>

[11] Chu Clewell, B. Breaking the barriers: The critical middle school years**.** In *The Jossey-Bass Reader on Gender in Education*. Jossey-Bass, San Francisco, CA, USA (2002), 301-313.

[12] Crawford, C. *The Art of Computer Game Design*. Available Online:<http://www.mindsim.com/MindSim/Corporate/artCGD.pdf> (1982).

[13] Druin, A. Cooperative inquiry: Developing new technologies for children with children. In *Proc. CHI 1999*. ACM Press (1999), 592-599.

[14] Farnham, S., Zaner, M., Cheng, L. *Supporting Sociability in a Shared Browser.* In *Proceedings of Interact Conference*, Tokyo, Japan (July 2001).

[15] Flanagan, M. Next Level: Women's Digital Activism through Gaming. In A. Morrison, G. Liestøl & T. Rasmussen (eds.), *Digital Media Revisited*. MIT Press, Cambridge, MA, USA, (2003), 359 - 388.

[16] Fogg, B.J., & Tseng, H. The elements of computer credibility. In *Proc CHI 1999*, ACM Press (1999), 80-87.

[17] Freeman-Benson, B., & Borning, A. YP and urban simulation: Applying an agile programming methodology in a politically tempestuous domain. In *Proc. 2003 Agile Dev. Conf.*, Salt Lake City, Utah (June 2003), 2-11**.**

[18] Friedman, B. Value-sensitive design. In *Interactions*. 3:6 (1996), 17-23.

[19] Friedman, B., Howe, D.C., & Felten, E. Informed consent in the Mozilla browser: Implementing value-sensitive design. In *Proc. of 35th Annual Hawaii Intl. Conf. on System Sciences* (Jan 2002), 247.

[20] Friedman, B., & Kahn, P. Human values, ethics, and design. In *The human-computer interaction handbook*. Erlbaum, Mahwah, NJ, USA (2002), 1177-1201.

[21] Friedman, B., Kahn, P. & Borning, A. *Value Sensitive Design: Theory and methods.* Technical Report 02-12-01, Comp. Sci. & E., UW, Seattle, WA, USA (2002). Available Online: <http://www.urbansim.org/papers>

[22] Friedman, B., Kahn, P. & Howe, D.C. Trust online. *Commun. ACM,* 43:12 (2000), 34-40.

[23] Friedman, B., & Nissenbaum, H. Bias in computer systems. *ACM Transactions on Information Systems* (TOIS). 14:3 (1996), 330 – 347.

[24] Glass, R. L. *Facts and Fallacies of Software Engineering.* Addison Wesley, Reading, MA, USA, (2000).

[25] Greenbaum, J., & Kyng, M. *Design at work: Cooperative design of computer systems*. Erlbaum, Mahwah, NJ, USA (1991).

[26] Grinter, R. & Palen, L. Instant messaging in teen life. In *Proc. 2002 Computer Supported Cooperative Work* (CSCW '02), New Orleans, LA (2002) 21 - 30.

[27] Hughes, T. *Human-built world: how to think about technology and culture*. University of Chicago, Chicago, IL, USA (2004).

[28] Inkpen, K., Booth, K.S., Klawe, M. & Upitis, R. Playing together beats playing apart, especially for girls. In *Proc. of Computer Support for Collaborative Learning* (CSCL '95), Bloomington, IN, USA (1995), 177-181.

[29] Kafai, Y.B. *Minds in Play: Computer Game Design as a Context for Children's Learning*. Erlbaum Associates, Hillsdale, N.J., USA, (1995).

[30] Kirkup, G. & Abbot, J. *The Gender Gap. A Gender Analysis of the 1996 Computing Access Survey*. PLUM Paper #80. (Programme on Learner Use of Media) The Open University: Milton Keynes, UK (1997).

[31] Latour, B. Where Are the Missing Masses? The Sociology of a Few Mundane Artifacts. In W. Bijker and J. Law (eds.) *Shaping Technology/ Building Society*. MIT Press, Cambridge, MA, USA (1992) 225-258.

[32] Laurel, B. *The Utopian Entrepreneur*. MIT Press, Cambridge, MA, USA (2001).

[33] Mubireek, K.A. Gender-oriented vs. gender neutral computer games in education. Dissertation, Ohio State University, Educational Policy and Leadership (2003).

[34] Muller, M.J., & Kuhn, S. (eds.) *Commun. ACM*, Special issue on participatory design. 36:6 (June 1993).

[35] Mumford, L. Authoritarian and Democratic Technics. In E. Katz, A. Light, & W. Thompson (eds.) *Controlling Technology: Contemporary Issues,* 2nd ed., Prometheus Books, NY, USA (2003).

[36] Nielsen, J. Paper versus computer implementations as mockup scenarios for heuristic evaluation. In *Proc. IFIP TC13 Third International Conf. on Human-Computer Interaction* (1990), 315 – 320.

[37] Nissenbaum, H. Values in the design of computer systems. In *Computers in Society* (1998), 38-39.

[38] Palen, L., & Grudin, J. Discretionary adoption of group support software: Lessons from calendar applications. In B. E. Munkvoldv(ed.) *Implementing Collaboration Technologies in Industry*. Springer Verlag, London,UK (2002).

[39] Papert, S. *The Children's Machine: Rethinking School in the Age of the Computer*. Basic Books, NY, USA (1993).

[40] Resnick, M. Rethinking learning in the digital age. In *The Global Information Technology Report 2001-02: Readiness for the Networked World* (2002). Available Online:<http://www.cid.harvard.edu/cr/pdf/gitrr2002_ch03.pdf

[41] Rettig, M. Prototyping for tiny fingers. In *Commun. ACM* (April 1994) 21-7.

[42] Richardson, H. S. *Practical Reasoning about Final Ends*, Cambridge University Press, Cambridge, MA, USA (1994).

[43] Salen, K. & Zimmerman, E. *Rules of Play*. MIT Press, Cambridge, MA, USA (2003).

[44] Schön, D. *The Reflective Practitioner*. Basic Books, NY, USA (1983).

[45] Shannon, V. The end user: Playing to a new audience. *International Herald Tribune* (11 JUNE 2004), http://www.iht.com/articles/524535.html

[46] Shneiderman, B. Universal usability. In *Commun. ACM, 43*:5 (2000), 84-91

[47] Soloway, E., Guzdial, M., & Hay, K. Learner-centered design. In *Interactions of the ACM*. 1:2 (1996), 37-48.

[48] Suchman, L. Do categories have politics? The language/action perspective reconsidered. In *CSCW Journal, 2*:3 (1994), 177-190.

[49] Von Prummer, C. Women-friendly perspectives in distance education. In *Open Learning*, 9:1 (1994), 3-12.

[50] Winner, L. Do Artifacts Have Politics? In *The Whale and the Reactor: A Search for Limits in an Age of High Technology*. University of Chicago Press, Chicago, IL, USA (1986), 19-39.

[51] Zimmerman, E. Play as research: the iterative design process. In B. Laurel (ed.) *Design Research*. Cambridge: MIT Press (2003). Available Online: <http://www.gmlb.com/articles/iterativedesign.html>