Buy Music, Make Money:
An Incentive-based System for Digital Music Distribution

Hannes Hesse, Kevin Lim, Yiming Liu
School of Information, UC Berkeley
{hannes, k7lim, yliu}@ischool.berkeley.edu

MIMS Final Project Report
May 2008

Keywords: digital music, incentives, economics

Abstract

We describe a digital music distribution system which fixes current problems in prevailing economic models for selling digital music. Our system aims to address the perverse incentives in the current digital music market, in which music buyers and music creators have incompatible goals and desired outcomes. The model gives users an added buying incentive: when a user buys a song, that user has a stake in how well that song does in the future. If a user buys a song, he or she is allocated a portion of the revenue from every future sale of that song. We present a proof-of-concept prototype of the system, to assess the viability of these ideas via simulated market conditions and via direct user interaction.

Contents

1 Introduction

We begin with a simple observation: the music industry is broken. Sales of music in the United States via physical channels have declined fairly consistently over the last decade [14], culminating in a 19.1 percent drop at the end of 2007 [11]. Even with digital music sales added, the total market has declined 11.8 percent, to a total value of $10.37 billion. This trend is consistent internationally, with the International Federation of the Phonographic Industry reporting a 10 percent fall in overall sales of recorded music in 2007, despite a 15 percent gain in digital music sales in that same period [13].

Many indicators suggest that many music listeners are getting their music outside the channel of commercial sales, through digital filesharing [10, 15]. Headlines about the industry pertain to lawsuits against fans and job losses at major labels [1, 3].

Much of the hype surrounding the issue of digital music is framed in moral terms. Opposing interests accuse the one another of being morally wrong, and try to use inflammatory rhetoric to dissuade others from taking actions that harm them. On the one hand, some music industry stakeholders posit that peer-to-peer filesharing is immoral, equivalent to stealing. On the other, many consumers assert that applying digital locks to music and suing fans are bad or oppressive actions. Despite legal implications, music filesharing is associated with a 'Robin Hood' aura of egalitarian justice.

Through the lens of economics, however, the fundamental problem can be reframed more objectively. We find that prevailing models for selling music suffer from misaligned incentives. Fans and producers are on opposite sides in the current model. Whatever the eventual rationalizations may be, it is economically sound for a buyer to prefer a free copy of music over one with cost. Given the minimal marginal cost of digital distribution and the wide availability of distribution tools, the music fan has every incentive to seek out free music online and share songs with friends. After all, it is a rational choice to prefer a free song over a $10 album or even a 99¢ single.

On the other hand, music creators derive income from the sales of songs, and thus have every incentive to stop fans from sharing music - from imposing draconian 'digital rights management' schemes to lobbying for direct government intervention in the market to preserve their business model.

At this impasse, it seems that innovation in the distribution model can provide a solution. Fans presumably enjoy music and wish to see the continued production of music, while content creators presumably enjoy the activity of music production and the financial rewards that allow them to continue. There exists a natural interest on both sides to see each other prosper, but misalignment of social and economic incentives prove a significant temptation from mutual cooperation. If some of these incentives can be aligned with personal interests on both sides, then a fair portion of the digital music problem can be effectively addressed for the benefit of all concerned.

In this paper, we propose an online music distribution system that corrects some of the incentive issues inherent in current models. We devise an economic incentive model that gives fans a stake in their favorite music and how it performs on the market, and describe a prototype system that implements this model. We examine the potential for this system to identify and reward musical trendspotters and trendsetters, and social and business implications thereof. The system is further tested for economic feasibility via a simulation framework, and for usability with a set of user studies. While much work remains to be done, we believe our incentive-driven distribution model for music will provide an effective answer to the problem of selling digital music.

2 Economic Model

2.1 Context

The throng of peer-to-peer filesharers reflect economically rational behavior, given available options and corresponding incentives. The typical price point of a digital music track as of 2008 is approximately $1.00, and the file may be encumbered by various digital rights management schemes that limit copying, CD burning, or even playing on various digital music players. Obtaining a copy of the same track, at similar quality, is free and usually a few clicks away from a filesharing site or peer-to-peer client. As the Internet becomes ever more user-friendly and well-indexed, the cost of filesharing – mental, temporal, and financial – have decreased.

Thus, consumers have little incentive to buy music from a record store, when they can download it for free from the Web or via digital file sharing tools. Further, consumers also have little incentive to stimulate music buying in others, when they can garner rewards by distributing free copies of tracks to others. There is social incentive in earning friends' gratitude and community recognition for providing others with low-cost music. In some cases, establishing a reputation of being an informed person – ahead of musical trends – are also factors that contribute to this social incentive. Filesharing cultures also reward those who share greatly with greater access privileges [2], which provides further encouragement to sharers.

Music industry stakeholders have acted – with good reason, given the threat to their business model – to maximize their own economic outcome. They slap the wrists of downloaders in various lawsuits, and implement schemes to control music files because they perceive rampant downloading as harmful to their revenues. However, attempts to shut down file sharing have largely been in vain, because new sharing sites and innovative filesharing technologies inevitably crop up from the ashes of old, destroyed ones. Further, switching costs for downloaders to move new sharing venues are low, since the value of the product (the track) is independent of the distribution site. It costs very little to go to a new web site or install new software. The burden to set up a peer-to-peer network, for example, is distributed across its beneficiaries [8].

The context of the problem comes down to a balance of intrinsic motivation and explicit value motivation [12]. The intrinsic, moral rewards for buying music through mainstream channels ("it pays the artists, I'm doing something good, I'm obeying the law") is insufficient for many people, because this does not outweigh the price for the music. We see a need to lay out heavier extrinsic incentives to encourage buying behavior, while not crowding out those intrinsic "good feelings" of paying artists – and staying on the good side of the law.

2.2 Existing Approaches

Big players in the online music market (Amazon, iTunes) seem to do well in terms of sales volume and major label visibility [5]. We point out, however, that they have essentially ported the traditional record store business model to the Internet. They have moved from physical CDs to digital downloads, but not offered a fix for the old record store model, which has seen the shutdown of big physical players such as Tower Records. The record industry seems to want to play the game of "shift the medium" once again, and is not recognizing these fundamental economic shifts, given the move from physical to digital distribution.

There are some innovations that seek to improve current models. One such innovation is the "referral program" or "affiliate model', which rewards buyers who refer their peers back to the same store. In some cases, a buyer must give a personal buying code to friends for referral tracking. In other cases, buyers provide the contact information of others.

There are hidden costs to these programs, which prevent these from becoming prevalent. Referrals require work on part of the participant; in some contexts, the rewards for this work are insufficient motivation. Second, there is a cost in nagging peers to join, potentially straining friendships and social ties. Existing referral models show that these programs reward those who can build large affiliate networks [9]. Most people aren't willing to do work or persuade others, just to receive some small fractional financial return on music. Nevertheless, there is a powerful potential benefit from such models: they align the interests of music producers and music consumers by giving buyers a stake in how well a song sells, which can create a deeper sense of ownership.

Another innovation is that of dynamic pricing. A song starts out very cheap, and price rise as song sales increase, reflecting higher demand. This is an innovative approach in that it incentivizes early buying by reducing the cost. In so doing, dynamic pricing models can create "momentum" for a given song, where buyers are encouraged to take chances on early music. However, these models eventually increase their prices to the standard rate of a song (about a dollar), so their benefits only last for a limited time, and raising the prices may have a damping effect on future sales.

2.3 Hypothesized effects

With alignment of incentives between buyers and producers, there is increased chance for users to buy a song. Further, buyers have an incentive against sharing songs for free, because that would depress their future rewards. Buyers have an incentive to encourage future buying behavior.

Our model creates a two-sided market and has a network effect: the more buyers, the merrier; the sooner (the purchase), the better. Trendspotters (buyers who predict hits) and trendsetters (buyers who can influence others to buy) are naturally rewarded and continue coming back. These power users are easily identified by this model, and special rewards may be allocated to these individuals. Both 'fame' and 'fortune' are possible within this model.

We accomplish this without complicating or changing the way people are used to buying music online. A typical user may find a song and click "buy" with no further action necessary. If he picked a hit, future rewards will begin to roll in whether he actively encourages purchases or not.

3 Prototype Implementation

In proving our model, we sought to answer two questions – whether the core concept of financial incentives for music purchases made sense, and whether the simple economic model backing it can work well in various market contexts.

To this end, we constructed a prototype Web-based music store that operated under our profit redistribution model, with a catalog of songs drawn from the music discovery service `Last.fm`[1]. A user interface was provided for conventional web-browser based interaction with the prototype, and users were invited to try out the prototype in several studies.

At the same time, a simulation framework was constructed against internal APIs, allowing automated software agents to maintain user accounts, browse the catalog, buy music, and otherwise interact with the store much as a human consumer would. Pre-programmed agents with distinctive buying behaviors participated in fixed-round simulations, in which their individual and aggregate performance of the store were assessed against a baseline, flat-price iTunes-like model.

While completeness of the prototype was desirable, the core of the questions we sought to answer could

[1] `http://last.fm`

be addressed by a relatively simple web application supporting only the basic functionality in a music store. Songs are represented minimally, without a full set of metadata (such as genre, bitrate, play counts, etc). Also, no actual music need be served by the prototype, and no payment processing system need be integrated.

3.1 Data Model

The data model for this prototype consists of five basic entities: the `User`, the `Artist`, the `Song`, the `Purchase`, and the `Earning`.

The `User` represents a registered customer in the store, with attributes for username, first and last name, email address, and account balance. It is associated with the `Purchase` entity, in the form of a 'portfolio' of purchased songs.

The `Artist` represents a music creator. The `Artist` entity possesses a limited set of descriptive metadata attributes, such as name, bio, and a MySpace profile URI. It is also associated with `Song` entities, establishing a relation between the songs produced by this artist.

The `Song` represents a single music track sold within the prototype store. It has metadata attributes for a title, length, year of production. It also has system attributes such as a URI to the actual music file, a price, and a split. The price is, of course, the price at which the song is being sold. The split field records the portion of the price available for reallocation to purchasers. The intent of this is to allow variable or artist-negotiable split ratios, which defaults to 0.5, or half of the selling price. It is associated with an artist.

The `Purchase` represents a sale of a song track to a given user. It is associated with a `Song` (the song sold) and a `User` (the purchaser). Further, it records sales transaction data such as the time of purchase and the buy-in position necessary for incentive redistribution. It is also associated with a number of `Earnings` entities (or so the purchaser hopes), which represents redistributed slices of the purchase price.

The `Earning` represents a redistributed slice. It contains a slice attribute representing the portion of the purchase price allocated to this buyer by the allocation engine. It is associated with a `Purchase` entity that represents the buy-in that benefits from this slice, another `Purchase` entity that represents the buy-in that created this slice.

For pragmatic purposes of effectively identifying individual records and for compatibility with object-relational management systems of popular web frameworks, all entities were assigned unique integer IDs as surrogate primary keys.

These five entities, taken in all, is capable of supporting the basic set of operations (browsing, buying, checking on personal performance) for interacting with the music store.

3.2 Architecture

The proof-of-concept system is implemented in Python, using the Django web development framework[2]. Many features offered by Django, especially its URI dispatch / routing layer, its database objects abstraction layer, and its UI templating capabilities were very attractive for rapid web prototyping. Further, as a Model-View-Controller web application framework [6] , Django enforces a separation of data, business logic, and representation, which creates a more elegant and maintainable application architecture.

The backend storage for the music store's state is a SQLite database. SQLite was chosen for its lightweight implementation of a relational database, its ease of use and embedding (no server required) within our system, and the easy transportability of its database files. By moving or copying SQLite database files into or out of

[2] `http://djangoproject.com`

our application directory, we can change the entire state of the deployed store. With pre-populated SQLite database files, we can have multiple store fixed at states of our choosing, ready to be swapped in or out for research, testing, and development purposes.

Two means of interaction with the store are possible: a conventional Web interface, and a internal Python API.

Figure 1 describes the overall system architecture for the music store prototype.

3.2.1 Web Interface

The web interface presents users with the basics of a digital music store, augmented with information and interfaces (described in detail in Section 4) specific to our incentive model. The web interface can be largely broken into components in three areas of functionality, following the classic MVC architectural pattern: the *models*, the *views*, and the *controllers*.

When using the system, the user interacts with a series of views of the store. The view is either a static representation of some set of resources and navigation links, or dynamically generated based on the user's context. The view comprises the UI, which is discussed in a later section.

When the user clicks on a link within a view, an HTTP request is made and a new interaction with the system truly begins. Based on the URI of the link followed by the user, the request is routed by the URI dispatch layer to the appropriate page controller. If the user clicks on a link for more information about an artist, for example, the request is handed to the Artist controller for a response. In Django, this is done using a series of regular expression-based routing rules, mapping URIs to their controllers and passing the necessary parameters.

The controller processes the request, extracts any pertinent parameters or request data, interacts with the data models, and oversees execution of the core business logic. In the above example, the Artist controller would use the data model to query for the specific artist being requested, retrieve pertinent information about the artist, and use a chosen display template to return the results to the user.

In a more complicated scenario, the controller may alter the state of the data, as allowed by the data models. If the request is to make a purchase, for example, the Purchase controller establishes the request, validates and deducts funds from the user account, and retrieves the link to the music file being sold. The slice allocation subsystem is used to determine the slice of future earnings due to this purchaser, and incentive slices are allocated to previous purchasers..

As a last step, the controller selects the appropriate HTML response template to the response, populates any necessary variables, and returns the filled-out template as a response to the access request.

The models are simply representations of the data in code. Models are used directly by Django's object-relational management subsystem to create and modify relational database tables and records.

The web interface largely follows a REST-based design [4], a best-practice for lightweight, extensible web application architecture. The music store's core entities use clean URIs as representations, while different types of HTTP requests serve as actions on these entities. For example, supposing the prototype store was hosted on `http://store.example.com`, a GET request to `http://store.example.com/artists/why` would retrieve all metadata information about the band "Why". A POST request to `http://store.example.com/purchases/`, with the artist name and song name or the store-assigned unique artist ID and unique song ID, would create a purchase for that song and associate it with the currently logged in user.

For the sake of simplicity and rapid development, certain elements of the store such as HTML-level user

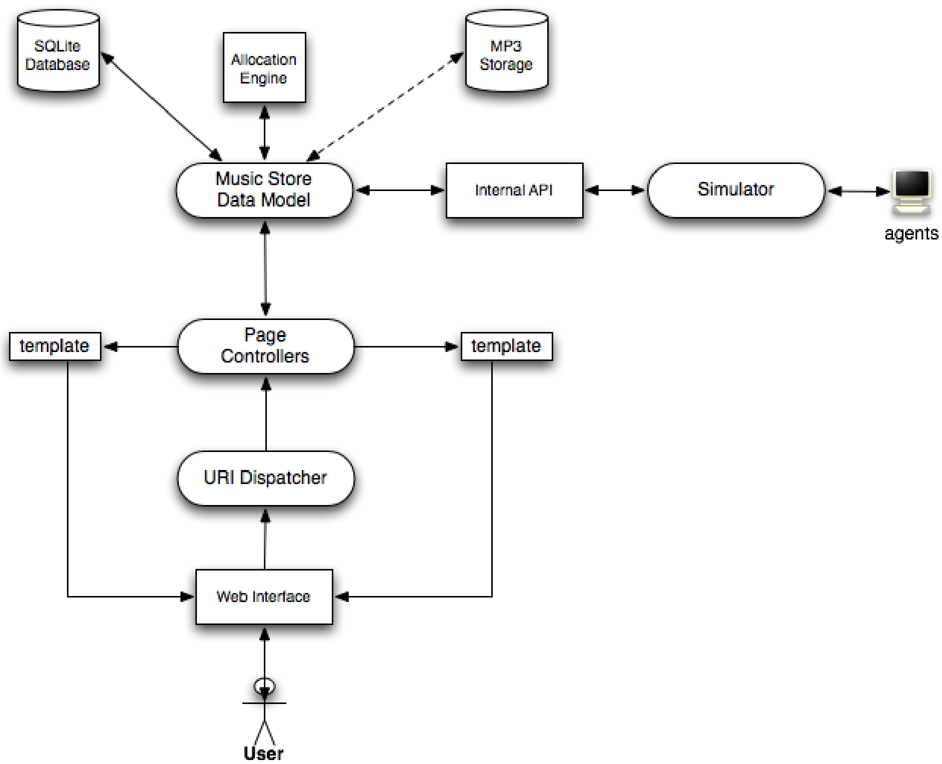

Figure 1: The system architecture of the prototype music store, with an internal data API for automated interaction and a user-facing web interface for conventional interaction.

authentication and session maintenance largely remain un-REST-ful.[3]

3.2.2 Internal API

The internal API consists of classes enabling direct access to the music store models, and a set of convenience functions that assist with automated interaction with the music store data. For the purposes of the economic simulation, the API exposed all five core entities directly as Python classes. Third-party code otherwise unrelated to the store can use these classes to create, retrieve, update, and delete entities in the store.

The immediate purpose of this internal API is to enable autonomous software agents, run in context of an economic simulator, to interact with the actual store catalog and browse/buy from the actual store database. Since automated simulations are a necessary component of this prototype store, the API allows reuse of existing business logic and internal software machinery to execute the simulation.

A second purpose of the API is to allow arbitrary scripting interface to the store. This enables us to push the store to a certain state, for example, by writing a simple Python script to buy a set of music, rather than dealing directly with the database or manually doing so from the Web interface.

A final, ulterior purpose is to assess the feasibility of an official store API, which can allow third-party developers to build on the store, its catalog, and its incentive allocation engine for their own purposes.

3.3 Simulation Framework

Using the internal API, the store economic simulation framework is designed to enable experiments with the store's catalog and allocation engine. The intent of simulations, with automated software agents, is to assess the feasibility of our incentive model under different market conditions and with different types of buyers.

The simulation framework consists of two major pieces: the `Buyer`, and the `Simulator`. The Buyer is a simple but customizable agent, representing a possible consumer in our music store, which makes purchasing decisions given a music catalog and a starting budget. The Simulator is an environment manager, which takes parameters, creates the simulation context, and coordinates the Buyer agents in the simulation.

As implemented, the generic Buyer is a simple agent with a budget and an incentive preference. The budget is a dollar-amount that the agent has, representing the amount of money the Buyer has and is willing to spend on music. The incentive preference, a real number ranging from 0 to 1 inclusive, represents the relative value the Buyer assigns to the potential of financial incentive as opposed to the actual value of the song itself. Accordingly, we refer to Buyers who have an incentive preference of 1 – that is, those who only care about a potential financial payoff – as 'gamblers'. We refer to Buyers who have an incentive preference of 0 – that is, those who only care about the intangible 'quality' of the song itself – as 'Music Lovers'. Of course, actual Buyer preference would lie somewhere in between these extremes.

The Simulator used for our experiments is an iterative one, with a fixed n number of rounds. At the start, the Simulator initializes s number of Buyers, either with fixed or randomized incentive preference values and either fixed or randomized starting budgets ($\$0 \leq b \leq \$100$). It also arranges all songs in the catalog according to a power law distribution[4], such that a few songs have a very high probability of being chosen from the catalog, while most songs sit along the 'long tail' of the probability curve and are less likely to be chosen. This is intended to reflect market reality of a few 'hit' songs amidst a large selection of undiscovered songs.

[3] It is also useful to note that Django's MVC implementation does not encourage REST-ful design in these cases. Building a REST-based session system, for example, would require overriding or ignoring the framework's built-in user and session-handling code. Doing so would impose a significant cost in "re-inventing the wheel".

[4] This distribution is fixed for the length of a simulation, but its shape can be changed as a parameter to the simulator.

At each round, the Simulator asks Buyers, one by one, to buy a song from the catalog. The Buyer evaluates the song first for its intrinsic value in dollar amounts, and then for its potential incentive payoff estimated as:

$$v_i = s * sl * w \tag{1}$$

where s is the network size, si is the allocated slice size, and w is a popularity weight on the song, as computed by the total copies of the song sold as a proportion of the total number of songs sold by the store as a whole.

Finally, the purchasing decision is made by computing the Buyer's personal value for the song:

$$p_p = v_i * P_i + v_q * (1 - P_i) \tag{2}$$

where v_i is the incentive payoff assigned to the song, v_q is the intrinsic 'quality' value of the song, and P_i is the Buyer's preference for incentives. If p_p is greater than the song's price p_s, then the deal is made the Buyer purchases the song. Otherwise, the Buyer skips this song and does not buy anything. Alternatively, if the Buyer's current budget b_p would be reduced below zero by the deduction of p_s from it, then the Buyer will skip this purchase round as well.

The simulator runs for n rounds, and for each round all s Buyers will be offered a purchasing turn. The Buyers are not aware of n, and thus we do not create any anticipation for the end-of-the-world in purchasing decisions.

Since the Simulator operates against the same data store as the actual store prototype, we can then visualize the results of a simulation run via the Web interface, interact with agents using human accounts, intervene to adjust the state of the store using the internal scripting API, and other perform other convenient actions.

4 User Interface

4.1 Overview

Designing a store like the one we propose can, among other disciplines, be framed as a user experience problem. A successful user interface for the system will facilitate a compelling browsing experience while providing the necessary information related to the economic model to make informed buying decisions.

We formalized the top-level requirements for the store user interface as follows:

- Encourage browsing the catalog and the exploration of new music
- Effectively convey the economic principle behind the store
- Establish trust in the brand and economic model
- Convey musical information as well as predictors of expected profits from the reward system
- Facilitate navigation through a potentially vast music database
- Encourage visitors to become customers by buying songs

4.2 User Study

We based our user study methodology on the one outlined in "Don't Make Me Think!" by Steve Krug [7]. Our philosophy was to test early, and to test often. A common mistake, according to Krug, is waiting to ask users basic questions which should ground the design from day one. Our user studies so far have come in two phases. First, prior to having a working prototype, we observed and conversed with users regarding our key competitors. Second, once we had implemented a basic working prototype, we tested the basic understanding of our site and performance of key tasks on our site.

4.2.1 Phase One: Competitive Analysis

We arranged sessions with users to use our competitor site. We were able to do this testing prior to having original designs, because we did not visit anything wed built. Instead, we used this testing session to ground our initial design decisions.

4.2.2 Setup

We conducted four one-hour sessions with different users, all UC Berkeley iSchool masters students, in a setting familiar to them, using their own computers. We asked each of them to walk us through the live site of a competitor. We tested two major online music sellers (Apple iTunes, Amazon.com MP3), a smaller, innovative seller of music (Amiestreet.com) and a source of social music recommendations (Last.fm). We received permission to videotape them for future review.

The key questions we asked here were:

- Do users understand the interface elements and functions?
- Which features or parts of the experience do they like, dislike, or love?
- Which elements work well and which do not?
- How do the applications and their features fit into users' lives and routines?
- How hard is it to do key tasks? (Key tasks involved exploring, discovering, rating, and buying music.)

4.2.3 Key Findings

- When competitors offered the chance to preview songs, every user did so prior to purchase. However, most users "skipped ahead" to sample different parts of a given song.
- One competitor was based around exploring other users' listening behaviors to find new music. These kinds of social features will likely contribute to a more compelling experience.
- One competitor required the user to leave their trusted domain while processing payment information. Besides disrupting the user experience flow, our test user found this discomforting. The user was inclined to terminate the transaction at this point because of security considerations.
- Some competitors provided a page with all previously purchased music as a personal library. This was well-received.

4.2.4 Phase Two: Testing Our Prototype

4.2.5 Setup

We conducted several more informal test sessions of our own prototype and hi-fi mockups. All test persons were UC Berkeley iSchool students. Krug advises to at such an early point not overvalue the truly diverse user base and recommends to rather conduct more sessions early than spending time and energy recruiting a representative test population.

The informal test sessions did not involve specific tasks. Instead, we exposed the test persons to the store prototype without much commentary and prompted them to think aloud, interpret all site elements and reflect on their experiences.

4.2.6 Key Findings

- Users felt that the store home page was too crowded and overloaded with store-related technical information. One user mentioned that it more resembled a stock trading site than a music portal.
- Users suggested an exploration of ”hotness” metrics (e.g. trend / avg growth over last x days). Show the hottest songs.
- Users were not convinced by the song profit table and linear graph. They doubted whether linear income would actually be possible with growing sales.
- Users did not immediately understand profit table, due to complicated grammar.
- Users felt that the site was missing primary navigation elements.
- Users felt the definition of song similarity was unclear.
- Users felt the need to focus on different parameters in different use-cases, and they requested song lists be sortable by multiple dimensions.
- Users wanted to favorite/bookmark songs (like starred items in GMail or watchlist on eBay).

5 Future work

5.1 Model and Systems

Much work remains to be done on both model and systems design. For the former, revenue model may require further modification to improve the profit margins for the store.

For the latter, scalability into an actual music store requires re-engineering of the backend architecture. While the ‘swappable’ SQLite store files and Django-driven prototype works well at low load, there are functional and performance drawbacks to their use. A scalable, deployable technology architecture for a highly visited music store needs a more robustly designed database abstraction.

To avoid the appearance of ‘empty shelves’ at the start of the music store’s launch, decentralized store widgets may be deployed on blogs, artist sites, MySpace profiles, focusing on a subset of the music being sold.

5.2 User Interface

Users encouraged us to continue developing towards the goal of a "fun" website that feels more like a music store than our current prototype. One obvious next step in this regard would be to allow a test user to play actual music as part of the experience. Moving away from the stock-market tone of the site was a theme of our user study, and we would like to increase the role of the music in our store.

6 Conclusions

In this report, we have described a novel incentive model for distributing digital content online that aligns the interests of all stakeholders. We envision a marketplace where new music is discovered by enthusiastic users. All stakeholders profit from our model, unlike many of the existing solutions in this space. Our results show significant promise, and we will continue to develop this work.

7 Acknowledgements

We would like to thank the following persons for for their support and advice: Prof. Yale Braunstein, Prof. John Chuang, Prof. Bob Glushko, Prof. Hal Varian, Fred von Lohmann, Jason Schultz, Nathan Agrin, Neal Fultz, Joseph Lorenzo Hall, Jon Hicks, Aylin Selcukoglu, Ashkan Soltani, and David Thaw.

References

[1] EMI reportedly cutting 2,000 jobs. *The International Herald Tribune*, 2008. International Herald Tribune, Reuters, The Associated Press.

[2] Nazareno Andrade, Miranda Mowbray, Aliandro Lima, Gustavo Wagner, and Matei Ripeanu. Influences on cooperation in BitTorrent communities. pages 111–115, Philadelphia, Pennsylvania, USA, 2005. ACM.

[3] Sudip Bhattacharjee, RamD. Gopal, Kaveepan Lertwachara, and JamesR. Marsden. Impact of Legal Threats on Online Music Sharing Activity: An Analysis of Music Industry Legal Actions. *The Journal of Law and Economics*, 49(1):91–114, 2006.

[4] Roy T. Fielding. *Architectural Styles and the Design of Network-based Software Architectures*. PhD thesis, University of California, Irvine, 2000.

[5] Saul Hansell. Amazon Gains Share of Shrinking Paid Music Market. *New York Times Blog*, April 2008.

[6] Glenn E. Krasner and Stephen T. Pope. A cookbook for using the model-view controller user interface paradigm in Smalltalk-80. *J. Object Oriented Program.*, 1(3):26–49, 1988.

[7] Steve Krug. *Don't Make Me Think! A Common Sense Approach to Web Usability*. New Riders Press, 1st edition, October 2000.

[8] Arnaud Legout, Nikitas Liogkas, Eddie Kohler, and Lixia Zhang. Clustering and sharing incentives in BitTorrent systems. *SIGMETRICS Perform. Eval. Rev.*, 35:301–312, 2007.

[9] Barak Libai, Eyal Biyalogorsky, and Eitan Gerstner. Setting Referral Fees in Affiliate Marketing. *Journal of Service Research*, 5:303–315, May 2003.

[10] Stan J. Liebowitz. How Reliable is the Oberholzer-Gee and Strumpf Paper on File-Sharing? *SSRN eLibrary*, 2007.

[11] Recording Industry Association of America. 2007 Year-End Shipment Statistics. Online, April 2008. http://76.74.24.142/81128FFD-028F-282E-1CE5-FDBF16A46388.pdf.

[12] Margit Osterloh and Bruno S. Frey. Motivation, Knowledge Transfer, and Organizational Forms. *SSRN eLibrary*, 1999.

[13] Eric Pfanner. Digital Music Sales Grow, but at Slower Rate. *New York Times*, Jan 25 2008.

[14] Ethan Smith. Sales of Music, Long in Decline, Plunge Sharply - Rise in Downloading Fails to Boost Industry; A Retailing Shakeout. *Wall Street Journal*, March 2007.

[15] K. Strumpf and F. Oberholzer. The Effect of File Sharing on Record Sales: An Empirical Analysis.