between Clouds & applications

“Finding new ways to interact with sound and data on the web”

Marco Hamersma
February - June 2011
SoundCloud Ltd.
After defining three different types of integration I analysed interesting uses of sound and data on the internet. An important conclusion was that a lot can happen after something is published on a platform. I noticed that a lot of data — or “objects” — go through similar processes: they get created, published, shared and commented upon. New objects can be created that have a relationship to the original, either explicitly or implicitly. I created a model in order to map the common processes and interactions upon objects on the internet.

The model shows that a lot of metadata is added after the initial upload: comments, ratings, tags, etc. The problem is that keeping this data synchronised between different occurrences of an object is hard. A solution could be an “ecosystem” of linked servers that synchronises metadata to and from all the occurrences of objects.

In the main case I explore the concept of object relationships for SoundCloud. I defined four types of relationships for the platform after which I created concepts to improve the user experience for these different types of relationships. Remix and sample relationships can increase discoverability for remixed and sampled content, whilst giving SoundCloud insight in how content is used. I also explore the concept of audio comments. Because tracks can be related to all kinds of content, object relationships would allow users to speak up on any kind of content around the web — tracks, videos, blogposts, images, movies, etc.

Eventually I give a couple of recommendations to how SoundCloud could expand on the concepts I described in the thesis. It features the idea that SoundCloud could collaborate with application developers to create a universal format for audio session files.
Preface

Moving to Berlin to work at a company like SoundCloud is an incredible experience, especially combined with working on your graduation project. I have had an amazing time in which I learned so much. Of course, sometimes it was hard. Moving to another country can be quite a switch and keeping everyone informed and getting feedback proves to be a challenge when you are a bit isolated like that.. This all proved to be a good stimulant for experimenting with ways to write parts of my thesis and presenting it in order to get feedback.

My thesis touches a lot of disciplines and I had a lot of ideas to start with, so it took me some effort to find a good course for the project and to keep focussed on the main case. Sometimes I had no troubles writing, other times I felt like I had bitten off more than I could chew. It’s been a very interesting time diving into these ideas and I’m glad to see how all pieces have fallen into place in the end.

I would like to give out a word of thanks to...

- **Hans Kemp**, for coaching me throughout the past four months. When I didn’t know how to proceed, a conversation with Hans was often all I needed to get back on track.
- **Roel van der Ven**, for helping me get started in Berlin & SoundCloud, giving me advice, guidance and helping me focus when necessary.
- **Jasper Schelling**, The conversations that I had with him in the months leading up to my internship were invaluable in shaping my initial ideas and during the internship he helped me get an insight that proved to be a catalyst for the rest of the thesis.
- **My family and friends**, who I was able to share and exchange experiences with during my stay here. That has been very helpful, especially during the more stressful moments.
- **The awesome people at SoundCloud** that I was able to work with, learn from and have a good time with. It’s been a blast!

Colophon

**Author:** Marco Hamersma  
**Student #:** 0793552  
**Email:** marco@marcohamersma.nl  
**Website:** http://marcohamersma.nl  
**Institute:** Rotterdam University of Applied Sciences  
**Programme:** Communication & Multimedia Design  
**Minor:** User Experience Design  
**Coordinators:** Hans Kemp, Jasper Schelling  
**Company:** SoundCloud Ltd., Berlin  
**Coordinator:** Roel van der Ven
# Table of Contents

Intro ......................................................... 5

 Glossary .................................................. 6

The definition of integration ......................... 8

Analysis of sound and data on the internet ........ 11

 Analysis conclusions .................................. 17

Defining an object interaction model .............. 19

 The Object Ecosystem .................................. 21

 Applying personal object servers ................. 27

Case: Object relationships for SoundCloud ...... 30

 User roles in social media ............................ 30

The model and SoundCloud’s different user roles 31

Types of object relationships ....................... 32

Audio Comments ......................................... 36

Remix button ............................................ 38

Sample workflow ....................................... 39

How to implement object relationships .......... 41

Case: Business aspects ................................. 44

SWOT Analysis .......................................... 44

Business aspects of my concepts .................. 44

Conclusions and Opportunities ..................... 48

 The answer to the research question ............... 49

Reference list ............................................ 50

Appendix: The Model .................................... 51

Appendix: Wireframes ................................. 52
Intro

About the Thesis
The internet is playing an increasingly bigger role in our daily lives, people are constantly accessing, storing and sharing online data. This dependency also creates the challenge of making these services and their data more accessible for users. While looking at the trend of cloud computing and cloud storage, I was getting more and more interested in how data from the internet would be integrated into applications. I’ve always had a huge interest in audio and sound, so that seemed like a perfect reason to dive into integration with sound data.

I love the idea of linking multiple services together, and I often think “Wouldn’t it be great if this data was integrated with …?” In my thesis I hope to find out in what ways data integration adds value for user experiences, after which I will find ways in which data integration between SoundCloud and other applications can be improved. With that knowledge I’m hoping to develop innovative concepts for SoundCloud and other web applications.

Although I will be addressing a couple of technical aspects, my primary focus is at the user experience of the concepts. The target audience for my research project is designers and developers of applications & web platforms. I hope my findings will be of use in developing better integrations that benefit the user experience.

Questions
To aid in the research I have formulated a central question to base the research on:

How can sound data exchange between an online audio distribution platform and applications improve the user experience and create new business opportunities?

This question can be broken up into a number of subquestions. The goal of these subquestions is to get specific answers that will aid me in finishing my research and the project.

- What are interesting uses of sound and data integration on the internet?
- What current & future examples can be given of data integration between applications?
- What opportunities for richer integration are there for SoundCloud?
- How can SoundCloud react to these opportunities?

About SoundCloud
SoundCloud is an online platform and community for people that want to share all kinds of sounds, they consider themselves to be the “YouTube for sound”. The platform is used by both
audio enthusiasts and professional artists and has been gaining popularity since its start in 2007. On June 15 2011 SoundCloud announced¹ that they had reached five million users.

Instead of focusing on the consumption of the content (like YouTube or Spotify), SoundCloud focuses primarily on the creators, facilitating them in sharing their tracks. Brand experience is an important issue with these users, but the brand doesn’t have to be very visible for consuming users.

SoundCloud is constantly looking for ways to increase the reach of its platform in order to stay the number one place where people share sound online. They want to focus primarily on their own product, but they are willing facilitate third parties to expand their reach. In order to do this they communicate with local artists, labels and companies through their offices in London & San Francisco. They present business propositions to those parties and can hack a prototype on the fly to give insight into what is possible with SoundCloud. It is SoundClouds mission to unmute the web and make it possible for anyone to share any kind of sound.

**Glossary**

There are a couple of terms that I use a lot in my thesis, here is a list of common terms and what the definition of those are.

**Object**

In my thesis I am using the term “object” a lot, I am not talking about real life objects but rather of an abstraction that is prominent in a lot of web applications nowadays. Most successful social networks are built around objects that connect people (photos, links, events, sounds, etc...)².

Each object has a number of attributes and data associated with it. Popular “objects” are video’s on YouTube or a track on SoundCloud. A track – for example – has an author, title, upload date, description, the audio file and it also has references to other objects (like comments). Facebook’s [API documentation](http://developers.facebook.com/docs/) lists 21 objects that are used on the site: among them are events, photos, status messages, links, users, videos, checkins and more. As you can imagine, objects have multiple attributes and associations with other objects, yet they follow a similar structure per object type.

Let’s look at another object to explain this object-centred thinking further: A movie. Like other objects, it has a standard set of attributes but can also include and reference other objects. A movie on itself has data like a title and a release date, but it also has *people* playing in it, it features locations, has trailers or video clips, memorable quotes and there might be ratings or comments added to it. These comments then have people who wrote them, etc.

¹ “*SoundCloud, an Audio-Sharing Site, Hits 5 Million Users*” — New York Times, June 15 2011

² “*Why some social network services work and others don’t — Or: the case for object-centred sociality*”, Jyri Engeström — Founder of Ditto, Co-Founder of Jaiku
Instances

Another term you’ll read a lot in my thesis is the term “instance”. I think it’s a difficult concept to explain, but I see it as this: “An instance is a manifestation of an object — In a way it’s a copy, but it’s a copy with its own attributes and associated data that doesn’t have to be the same as the original”. To quote an example that is commonly used when explaining object oriented programming: A car is an object that has a couple of standard attributes, yet your car (the instance in this case) can have different values for those attributes (like the colour). A more concrete example in the context of my thesis is a embeddable widget from YouTube; a
The definition of integration

When I talk about “Integration” in my thesis, I’m mostly talking about sharing data from one service or application to another. This could be as simple as a widget from a service like Last.fm, but can also be a lot more elaborate like synchronising data from an audio app with a web service. There are many ways of integrating data, but they can be categorised into three levels:

1. Content aggregation

Getting content from one service and making it available inside an application or website. Think about RSS readers, or widgets that let you embed a Twitter timeline on your website.

Both of these examples get data from a web server and display it. In the case of the Twitter widget this is text, but it can also be data like weather as displayed in the weather widget.

Although an API is sometimes involved, this category often doesn't require a form of authentication and never involves actual writing access.

2. API Integration

This is about integrating functionality of the original platform in an application or website by utilising an API. Think about sharing content on Twitter or uploading to SoundCloud from a music making application. In contrast to the previous level the API is used to actually interact with the other service.
The iPad app Flipboard lets you consume content from sources like Facebook, Twitter, Google Reader and Instagram. It also integrates with the API’s of those services to allow you to interact with that content by liking or commenting.

3. Object Syncing

“Object Syncing” is about letting data services autonomously synchronise objects between different places. This is something you rarely see, but it has a lot of potential and I think it’s the next step in integrating data. This type of integration will be the main focus of my thesis. An example of this type of syncing can be found in Apple’s Aperture:
Aperture can integrate with Flickr and Facebook by displaying photo albums/sets inside the application. Instead of simply allowing you to push photo’s to an album you can alter the order of the photo’s inside a set. In the process Aperture keeps a link with the original photo, allowing you to change metadata (like title and tags) in one place, to have it synced back to the other. This type of integration is far from perfect, though. Making changes to the actual image leads to an unexpected result: the image being re-uploaded and the original — along with all it’s comments and favourites — being deleted, auch!

The three levels…

When you look at the different implementations and the examples that I’ve given for each level, you can see a change in technical complexity. With each level, the costs to implement increase. Object syncing — for example — is a terribly complex thing to do\(^3\), while offering a widget is in most cases quite easy since it is often existing functionality that is repackaged. Still, the complexity can be hidden or simplified a bit through layers of abstraction, both for the end user and the person implementing this integration. When using a Twitter widget — for example, the complexity is often not even visible for the end user, and the person implementing it can just use one line of code.

\(^3\) This article by Cultured Code describes a couple of the problems that arise when trying to create a reliable syncing system
Analysis of sound and data on the internet

In order to better understand how data (and sound data in particular) is used, I analysed popular and interesting uses of sound and data on the internet. I analysed why an example is interesting, how it works, how the usage effects different stakeholders and what happens with the data in these examples — primarily as an abstraction. Analysing what happens to the data is important in order to detect patterns that can be used in this thesis’ case.

Heapsong #1 / Lifeline

Heapsong #1 was a project by artist Imogen Heap:

"#heapsong1 is the working title of the first song from my new album. The album will be completed in roughly 3 years, with a new song released every 3 months as soon as it's completed. My fourth solo album begins here, with you on the 14th of March. You are the spark of inspiration. [...] I love the idea of turning the tables in that the seeds of the song begins with you, making a full circle when you experience it as a finished piece."

SoundCloud was used as the base for this collaboration project. Users could upload sounds in a certain window of time, after which Imogen would listen to the sounds to see which ones would be used in the song. Communication with her and the fans went through a video stream and a chat. She basically created a collaboration platform by connecting different services on the internet. These services get exposed to a group of potential new users, and because their “idol” uses those services, they are more likely to start using the service.

What is happening with the data

The collected sounds that are uploaded by users are all related in the fact that they are uploaded for this project by Imogen Heap. After the collection phase was over, Imogen loaded them into her audio application and started working on the track. Together with new elements the track was eventually uploaded back onto SoundCloud with a link back to the project’s website.

indabamusic.com

Indaba is an online platform for music collaboration. The website is specialised in stimulating collaboration between people and also offers musicians help in finding suitable projects based
on skill level, influences, instruments, location and previously recorded content. It uses popular and celebrity users to promote itself.

**What is happening with the data**
Content is uploaded by users, which is then downloaded by other users in order to use the samples in their own projects. The content is then published on indaba music or other platforms.

... *takes Questions*
A project that stems out of a collaboration of SoundCloud with Imogen Heap. It allows users to record questions which can then be answered by the person who owns the "... takes questions" page. It is an interesting concept because it doesn't use video but instead relies on audio. This makes it more accessible to people while still being richer than text. New users are able to quickly set up, style and share their own “takes questions” page.

**What is happening with the data**
The "takes questions" page is the platform on which all the content is displayed, but both the questions and answers are stored on SoundCloud. Somewhere in the back-end of the website, the newly recorded questions and answers are linked to each other and to the "takes questions" page that they belong to. In other words, the two tracks are related to each other.

**You are listening to...**
A website that combines audio streams from Police scanners with Ambient music from SoundCloud. This is a cool example of real-time interactive audio mashups. It is interesting because it a “mashup” using partially live audio. People find it inspiring, atmospheric, almost haunting. It works because it is something new (mashing up live audio) and people want to share it.

**What is happening with the data**
A curated list of tracks from SoundCloud are played, while in the meantime an audio stream of a police scanner plays.

**Gettyimages Moodstream**
Moodstream brings together audio, photo's and video footage that are tagged with moods, colours and other aspects. By manipulating set of mood-sliders the user can sit back and look & listen at a collection of media that suit the mood the user wants to achieve. This is possible because of a huge database of extensively tagged content. It aids users in the creative process and also offers related media that can be bought, which increases sales for Gettyimages. It is interesting because in this case the extra metadata that was available made it easy to use content for a purpose that it wasn't used for in the first place, it can open the door for all kinds of mashups.
What is happening with the data
The content that is displayed in the application is content coming from the database of GettyImages. Each item is extensively tagged with things like objects, activities, people, concepts, moods and style. These tags are then used to filter photos on the concepts that are used in the application: mood, warmth, nostalgia, calmness, etc. When a user changes a slider a new batch of media is selected from the database.

iDaft
A small app that allows you to recreate a couple of Daft Punk songs by letting you play with samples that were used in the song. It works because the songs are integrated in popular culture and the execution of the idea is simple. You just tap on the sample that you want to hear. The app is effectively a promotional platform for the artist.

What is happening with the data
The original song is split up between a "backing track" and the vocal samples that are used in the original. These samples can then be played in random order by the user.

NOS Media Cloud & BBC Fabric
These are examples of systems that are currently being developed by broadcasting companies in order to exchange media material (like audio and video) with local broadcasters or other third parties. This is interesting because it hasn’t really been done before. Most big broadcasters nowadays share some of their news content on the web, but this usually for consumption purposes (i.e, regular viewers), not for peers. This way broadcasters are able to benefit from each others work.

What is happening with the data
The technical details on these services are not public, so part of this speculation. The media from multiple parties is uploaded and centralised in a central service, after which it is tagged with relevant information. Other parties then have access to that content and can then re-use that content for their own means. In theory these parties can all add value to existing content by tagging or commenting, but it depends on the permissions within the system.

RTV Rijnmond
This is a regional broadcaster in the Netherlands that makes extensive use of SoundCloud as a way of storing and sharing interviews and reports. They do a similar thing with video using YouTube. RTV Rijnmond is a good example of a media company that managed to integrate online

4 Press releases related to NOS Media Cloud and BBC Fabric:
“NOS deelt materiaal met regionale omroepen via NOS Media Cloud” – Radiofreak.nl, 7 March 2011
“BBC websites cost users 67p per month” — The Guardian, 6 July 2010
media into a journalistic workflow. Interviews are recorded on the radio, published on SoundCloud and then embedded in related news articles on their website. It allows them to expand their reach and the quality of their online presence without having to spend a lot of money on infrastructure for storing audio.

**What is happening with the data**
The sound is recorded and then uploaded to SoundCloud where it is stored. It is then embedded in an article on the website of RTV Rijnmond, where it adds value to the article (again, the content is related). The article can then be commented upon by readers.

**KORG Apps**
KORG makes a couple of iPad apps that emulate analog synthesisers and samplers on the iPad. The apps have a very good integration with SoundCloud. Like more and more audio-creation apps it allows you to upload your creation to SoundCloud, but it also allows you to upload your session data with it, allowing other people to continue your work or learn from it. It uses content created by users as a source for new content. In the meantime, SoundCloud is promoted through the app.

**What is happening with the data**
New tracks get linked to the application that made them and when the user gives permission to upload the session file together with the track, that session file is then linked to the track object. Afterwards, users can search for tracks that have session files and can then re-use or remix them.

**Wikipedia**
Wikipedia is a web-based encyclopaedia that allows its articles to be edited by anyone, that is why it is constantly in flux. Its users are constantly refining and expanding the website’s content. Articles from Wikipedia are used in websites and applications everywhere — iPhone and iPad apps or sites like Qwiki and Facebook. The latter of which displays Wikipedia content to display content on "likeable" pages that are not yet claimed. Wikipedia proves that even small changes can add value to an article. Everyone can contribute a bit of knowledge to a specific topic.

**What is happening with the data**
After the creation of a page, users are allowed to make incremental changes to the articles. All the changes are then stored, even when they are overwritten by new changes.

**Qwiki**
A visual encyclopaedia that takes content from a number of sources (including Wikipedia) and automatically finds pictures of topics, people and companies that are named in the entries. It also
links to related content — both inline and after the entry has been played. It is an exciting new way of exploring the concept of an encyclopaedia.

**What is happening with the data**
Results for certain topics are gathered from multiple sources and then summarised by an algorithm that also finds related content and topics. It is important to realise that content is not "created" by Qwiki. It is dependant on other sources, but then re-uses the content, mixes it together with other content in order to add value.

**Calais & Zemanta**
Calais is an application that can read a piece of text and then adds metadata like tags, topics and names in the form of Linked data. Zemanta is a similar service that finds related articles, media, biographies, websites, tags and other metadata for a piece of text. It is interesting because it shows the power of linking data together and gives the end user access to all kinds of related information that can be used for enhancing an article.

**What is happening with the data**
The input data is processed by an algorithm that recognises topics and keywords. Calais then adds those topics and keywords to the original text in a machine-readable format that can then be easily used by other applications. Zemanta finds related media on the internet for those topics and keywords in your content.

**Flickr’s camera finder**
This is an interesting usage of metadata on a web service. Flickr stores the camera’s metadata with the photo and makes it accessible to users of the platform, they also analyse this data, which allows Flickr to display the most popular photo cameras. This whole thing gives both Flickr and its users insight in popular cameras and might help new users get started with photographing, this is where the real added value for users is.

**What is happening with the data**
The data is saved into the photo file by the camera. After the picture is uploaded, Flickr analyses it and saves the metadata that is then made available to users. It also uses it for statistics.

---

5 An article about the launch of Qwiki, talks a little bit about how it works:
“Disrupt Winner Qwiki Arrives In Private Alpha (1,000 Invites)” — TechCrunch, 22 October 2010
YouTube’s “As seen on” and Creative Commons integration

These are two new features on YouTube that do something interesting. “As seen on” is a feature that promotes websites that link or embed videos:

Every day, you embed YouTube videos on blogs and sites across the web, often adding their own commentary and perspective. This sharing of videos helps to show the variety of great content on YouTube, and often adds context to what you’re watching. [...] By crawling web feeds of sites that have embedded videos, we’ve built dedicated pages that highlight your embedded videos. [...] We think these pages provide a way to find new and interesting content while helping you dive deeper into the conversation around a video.

This is interesting because shows the places where a video was used. That way, the process of embedding works both ways: The place that links to the video adds value to an article, but viewers of the video on YouTube can visit the article to get extra information or context of the video. Slideshare does a similar thing by showing the places a presentation was embedded.

YouTube’s Creative Commons integration allows users to see source files that were used in a video that is made with YouTube's video editor. This data is then displayed beneath the video:

What is happening with the data

For “As seen on” YouTube analyses crawls websites for links to its video files and links this data to the video or presentation, effectively making the reference bidirectional. YouTube's Creative Commons integration works this way because there is a closed system (the YouTube video editor) which saves links to the source data.
Analysis Conclusions

In this chapter the conclusions from the analysis of sound and data on the internet are summarised in order to give meaning to the discoveries.

An interesting generic conclusion about audio is that it much richer than text, yet more spontaneous and accessible than video (you don’t have to clean up your room or do your hair before recording).

Goals for integrating content between applications:

There are a number of reasons why people would want to integrate their websites or applications with other applications.

A common reason is to add value to an existing product: an application wants to expand the data that is available to its user by implementing content from another – often more specialised – website. Think about Facebook using content from Wikipedia to fill their “pages” or SoundCloud showing tour dates from Songkick on artist profiles. The source of that data gets attributed which might drive more traffic towards their service, while the platform that included the content gives its users more relevant data.

The second most common reason for integrating applications is content hosting. In most cases this means that a website/app uses content platforms like SoundCloud or YouTube to get an easy, reliable and economic way of storing data. These platforms can enhance a user’s workflow by taking care of certain common processes like sharing and distribution. RTV Rijnmond is a good example of this. The broadcaster records radio interviews and video reports, publishes them on content platforms, which then allow them to easily embed the content in news articles on their website. The articles then serve as a touchpoint for the content platform, because of the usage of branded widgets.

Adding value

As the KORG apps and sites like Wikipedia and Indaba music show, a lot can happen after something is published on a platform: Other users can add value to it or re-use or remix it. It can be a continuous cycle. When a website links to an object, it is often because that object has added value to that website. In the case of a video, it takes the object and puts it into the context of an article, for example. YouTube and Slideshare illustrate that doing the opposite also adds value. They link back to the places that use their content. That allows users to learn more or to see the video in another context.

Another conclusion is that content can be related to other content because they add value to each other, re-use each others content or give context. For example, a video adds value to a news article. These relationships can be both explicit or implicit.
**Metadata**

Metadata is the key to allowing different uses with existing content, for example insight in tools (like Flickr) or a new way of discovering content on a platform: **Getty Images’ Moodstream** is an example of what can be done with a database of extensively tagged content. Another very good example of what is possible when adding metadata to files is **Mashup Breakdown**. The data in this example was manually added to the song for the website, but it helps to show the potential of what metadata can do.

**Hacks & Mashups**

Mashups and hacks are great ways to promote your service as a content platform. Just make sure that they are simple to use and that they can easily be shared on social networks. Audio mashups haven’t really been explored much on the internet, but they can offer a cool way of promoting audio content. **iDaft** is a cool example of letting users play with samples.
Defining an object interaction model

In the analysis of interesting uses of sound and data on the web, a pattern can be found in the different interactions that are performed on the objects. In a lot of cases the interactions are about adding value to existing objects. After an object is published, both people and applications can add extra context, meaning or value in the form of replies, re-usage, embedding and sharing. These are common processes that can be mapped in a model:

The model illustrates a common process that objects on the internet go through and defines three channels through which this process can be restarted:

1. **Contributing** to the meaning, value or context – often through comments or metadata
2. **Sharing the object** and republishing it – e.g., by publishing a widget on your blog.
3. **Remixing the object**: getting the objects contents and reusing or repurposing it, starting a new cycle.

Most media can be remixed, but not all other objects have all three of these “cycle channels”. for example, a "location" object cannot usually be remixed.

**The use of the model**

Because the model lists common processes that are performed on objects, it can be used to evaluate an application that uses objects. The different steps and cycle channels can be renamed to what's appropriate for an application after which you can discover opportunities for increasing the user experience for those common processes.
Testing the model

The model describes common processes that objects go through, but as described above, not all object types have the different cycle loops. These models describe the different cycle loops for the following object types: sound, people, locations, posts (news) and raw data:
The Object Ecosystem

In the previous chapters I have introduced the concept of standardised objects on the Internet. The model shows a number of common processes that these objects go through: people add value or context to them, share and distribute them or re-use its content to create something new. These processes happen all the time, but there is no way to make these processes visible unless they happen within a "walled garden"; like Tumblr.

Persistent Metadata

Once a post is reblogged outside of Tumblr, the link between the original and the copy is gone. New interactions that users perform on the copy are not visible on the original, while the original
could potentially benefit from this added value. The same applies if that content is then copied yet another time. A more concrete example is content from SoundCloud that is embedded on a music blog: comments and other metadata from that music blog are not visible on the track’s page on Soundcloud, while they are still relevant to the track object. In other words: this data, even though it adds value, is not persistent throughout all the different instances of that object.

Object syncing - as introduced in “The definition of integration” - would be the solution to this by allowing instances of an object to send and receive changes in metadata. This is what I call “Persistent Metadata”, as the metadata is now persistent over all the instances of an object. Object syncing would allow an “Object Ecosystem” — a system of linked standardised objects all across the internet that websites and applications can interact with.

This chapter will describe what the object ecosystem would look like and what benefits it might have for web services.

**Object Servers**

In order for objects to exchange data between all the different instances you need a central place that stores the object and sends and receives metadata to and from all the instances of that object⁶. In the model I explained that objects get published to a platform, this platform is the source in the object ecosystem.

---

⁶ The object ecosystem needs object servers because the alternative would require an object to communicate with every other instance directly — causing enormous amounts of traffic, logistically having a central server is more reliable.
At the moment, these platforms offer content through webpages or the API. In the object ecosystem this content could still be requested through an API, but the content would be offered in the form of objects. The webpages that display them will merely show instances of the object. When content is shared, a link is made between the object server and the new instance. This allows for the syncing of changes in metadata to the object.

In the object ecosystem, every website or application that produces content either uses an object server or is actually an object server itself (creator platforms and storage services, for example). Every object can be linked to, reused or — depending on permissions — edited.

Object services are not limited to one specific type of content. They can serve all kinds of objects. As mentioned earlier, Facebook lists 21 different kinds of objects in their API documentation. Some of these objects are collections of other objects. Photo albums – for example – have their own attributes, but they include references to a number of photo objects. The idea that objects can be inside “collections” is a key part of the object ecosystem.

Resources on object servers (personal and public)

In order for an object server to store objects, it needs to categorise it with similar objects. This collection is called a “resource” and this resource can be accessed by other applications. A service like SoundCloud will have a number of resources (tracks, users, sets) that are all public.

Not all content should be public, however. Some objects should only be visible after a form of authentication. Hidden tracks, private data or draft posts, for example. Also, some of these objects are allowed to be edited, others are not. That’s why there needs to be a way for Object Servers to be able to give certain parties access to data. This authentication can be fully automatic (e.g., through an API) or it could be manual.

Personal object servers

Not only websites and applications could have object servers, but also people. This is especially useful for centralising personal data so it can be accessed from multiple applications and devices (phones, TV’s, refrigerators). A good example of this is a list of contacts. Both your address book application and your phone might want to use your object server as a place to get that data.

Like with other object servers, not all data is public. That’s why applications need to request permission to access certain resources (like your contacts or your profile). An object server on a content platform can do this automatically, but with personal object servers this is done manually.

After setting up your personal object server, apps like the address book might ask: “Can I store your contacts on your server?”. Devices or applications can then ask you permission for access to that resource (or a subset of it). Next to simple reading access, applications can also ask for permission to write data to a resource. In some cases, applications might even want to create new resources on your object server that are accessible to other applications.
The operating system detects an object server and offers to store data on it.

Another advantage of personal object servers is that instead of having your data scattered across multiple servers, everything could be stored in the same place. An example of this is a “favourite videos” resource that video platforms like YouTube and Vimeo contribute to. The result is a centralised list of videos that you like. Other applications can then ask access to that data to do recommendations, for example.
Marco’s API is a concept that centralises all kinds of data about me and makes it available for other applications to use.

**Setup**

Because object servers work for all kinds of data, it only needs to be set up once. Routers would be an ideal device to host the object servers because they can automatically configure itself. This takes away the hassle of setting up a server and making it accessible from outside of your home. Website control panels could come with an object server preinstalled just like they come installed with support for programming languages.

Authentication for personal object servers will go through the operating system or the browser. Application and websites can request an object server to access a resource. The browser or operating system then gives the active user a notification or alert in which the user can authorise the service. After authentication has been granted, the user can withdraw it at any time, preventing the server from accessing the data. This gives users more control over what accesses their data.
When registering on a website via an object server, the browser asks the user to allow the website access to certain resources.

Decentralisation

Personal object servers are an interesting idea because it allows people to easily host their own data. There is a big debate going about what happens to our own data. Stephen Hay addresses this matter in a blogpost⁷:

For a while we’ve posted our data all over the internet on all types of services. These services provide APIs so we can access the data we put into them, so that we can do things with that data. Read that again. [...] What if we flipped this all on its head? What if we hosted our own data,

---

⁷ Death to web services. Long live web services! — 17 December 2010
and provided APIs for all these webapps so that they can use our data? Isn’t one of the points of the semantic web to make decentralised information meaningful, retrievable and mixable? We provide the structured data, Flickr provides the functionality. The sharing. The social. Why not? Services don’t need to host our data. They only need to do cool things with it. If I change my information, it’s automatically changed on all the services using it. Storage space is up to me. Privacy settings? Totally up to me.

In other places, big discussions are taking place and initiatives like the decentralised social network Diaspora* are becoming very popular. It has also spawned concepts that are similar to the vision of an object server as described in this thesis. For example the Personal Data Ecosystem, Project Nori and Unhosted.

These similar services are all built on the idea that you should host all your data, and this is an important difference with the concept that is described in this thesis. Most data could — and in my opinion should — be hosted yourself, but big files could still be hosted on content platforms like YouTube or SoundCloud for logistical reasons. The objects are still centralised, but it doesn’t matter where the data/files are hosted.

**Hosting all your own data**

The most important reason for making this difference is that hosting all your own data is not always realistic. Traffic is the biggest issue: viral videos – for example – can get viewed tens of millions of times. The average internet connection won’t be able to serve this. Like other media services like Flickr and SoundCloud, the data needs to be processed in order to be of use to anybody. As stated in “the definition of integration”, people use these content platforms so they do not have to take care of all the processing involved. If platforms aren’t allowed to store data anymore this would move the processing of media from a well equipped server to the creator or the end user. It probably won’t allow web services to do the "cool stuff" that Stephen Hay was referring to.

**Applying Personal Object Servers**

The previous chapter introduced the idea that metadata added to different instances of an object all get synchronised to each other through the object ecosystem: a network of linked object

---

8 A comment thread on the blog of Jeffrey Zeldman, again about hosting your own data: “Own Your Data” — 10 January 2011

9 People use content platforms so they don’t have to think about things like media transcoding. If people hosted their own data, people would be required to transcode their video to 10 different formats in order to store it on YouTube. This 2010 article by “Ad Terras Per Aspera” mentions eight different formats and omits two WebM varieties, which are included in this Wikipedia article. These are probably not all the video format YouTube transcodes in, but I couldn’t find other sources that suggested that. 10 is already a lot, and it proves the point.
servers that store all kinds of objects. It also introduced the idea of personal object servers, servers that store objects like contacts, music lists and locations in a place that the users themselves have control over. This chapter explores how services like SoundCloud would make use of these systems, and how it has added value compared to the current situation.

Object servers allow applications to get a complete user profile right from the start: Names, avatars, social networks, email addresses and maybe even friends, top artists, etc. For SoundCloud this would mean that it can recommend friends or artists to follow. The data would also be more recent and possibly more complete since the user maintains it for all kinds of services and syncing would make sure all applications receive the updated info automatically.

Using object servers might mean that profile data is already complete right after a user created an account. If applications are allowed access to a user’s address book and “top artists” resource, websites like SoundCloud could give better follow recommendations.

With Object Servers in place, applications can get easier access to data from other services and also allows multiple services to contribute to a shared resource. For example, applications like iTunes, Last.fm, SoundCloud and Spotify could all contribute to a list of favourite artists on a user’s object server. The result is a nice dataset, but it actually gets useful when applications and people can use this data. Applications might be able to improve their experience with this data and music bloggers will be able to show a list of favourite tracks on their blog.
John's music blog

New album
Far far away, behind the howl mountains, far from the countries Vaslavka and Connetable, there is a little village. The villagers live in Dubrovnik, a large language island.

An article
Far far away, behind the howl mountains, far from the countries Vaslavka and Connetable, there is a little village. The villagers live in Dubrovnik, a large language island.

A video
Far far away, behind the howl mountains, far from the countries Vaslavka and Connetable, there is a little village. The villagers live in Dubrovnik, a large language island.

This video is not available in your country.
Case: Object relationships for SoundCloud

In the chapter “analysis of sound and data on the internet” the concept of object relationships is introduced. In this chapter I explore concepts that utilise object relationships for SoundCloud.

User Roles In Social Media

In order to better understand how users could benefit from object relationships it is important to get an understanding what kind of user roles there are. The "Ladder 2010" model by Forrester Research describes seven roles of users. The model below (published on Business Week’s website) gives a more graphical representation of these user types. The vocal part (conversationals and critics) have been consolidated into one "critics" category:

Instead of referring to these categories as "types of user", It is better to refer to them as user roles since there is overlap and some users can be a mix of multiple roles. For example: it is likely that most creators are also critics.
Users on SoundCloud

I decided to use Forrester’s model as a foundation for defining the different user roles that are present on SoundCloud. There are three things that I went trough while converting Forrester’s model for Soundcloud.

1. Splitting up the creators into people that actually create content on SoundCloud and those that don’t. The latter category consists of "distributors" and "sharers" — people that write about content on SoundCloud or create content that uses content from SoundCloud (videos, for example).
2. Combine "critics" and "conversationalists" to form the vocal part of the community.
3. I thought it would be better to combine the "joiners" and "collectors" because having this role on SoundCloud implies an active use of SoundCloud and its features.

After Forrester’s model is converted for SoundCloud, there are 5 user roles for the platform:

1. **Audio Creators**: These users are the beating heart of SoundCloud. They are the users that create content on SoundCloud.
2. **Sharers**: This group describes the people that discover content on SoundCloud and share them by writing about them on social networks or blogs (i.e., places other than SoundCloud). In a way they are creators, but create content somewhere else. They have an influence on SoundCloud by sharing its content. A lot of the users in this category also belong to the Critics group — especially when their publishing platform is a music blog.
3. **Critics**: This is the vocal part of the SoundCloud community, the people that participate in the community by commenting and responding on content, instead of just listening to it.
4. **Consumers**: These are the people that are familiar with SoundCloud, have an account, follow people and know how to favourite items. These users know how to use the site effectively, but have a more passive role than the critics.
5. **Occasional Listeners**: These are visitors that don’t really use SoundCloud very often and occasionally encounter the website or its widget (on music blogs, for example...).

YouTube mentions similar user types in their blogpost "Inside User Research at YouTube" — They also speak of a group that focuses primarily on consuming content, a vocal group and a group that is focused primarily on creation. Again, there is quite some overlap between the creator role and the vocal part of the community.

### The Model And SoundCloud’s Different User Roles

In the previous chapter I defined 5 different user types on SoundCloud: *audio creators, sharers, critics, consumers & occasional listeners*. It is important to link these user types to the different stages of my model, for this I’m using a version of my model that’s tailored to SoundCloud:
Audio Creators
Audio creators’s primary activity in the model is the “creation” part, which also includes publishing the content on SoundCloud. Because they tend to be active users of SoundCloud, they are often also part of the “critics” group.

When looking at the ‘cycle channels’ in the model, creators are the primary users of the remixing channel.

Sharers
The most important roles for sharers is adding meaning to the objects by publishing SoundCloud’s content on places other than the initial platform (SoundCloud). They are active consumers (listeners) of the content and often also belong to the “critics” group.

Their primary cycle channel is the sharing channel.

Critics
The most important phases for this group are the last two: adding meaning and listening. The corresponding cycle channel – commenting – is also their primary channel.

Consumers
These users are almost exclusively involved with the listening phase. They do – however – add some value to the object by marking a track as a favourite.

Occasional Listeners
These users are only consumers of the content, so the listening phase is basically the only phase they are active in. They are not involved in any cycle channels.

Types Of Object Relationships
In my distribution model I discussed a number of common interactions and so-called “cycle channels” of objects on web services. On SoundCloud, all the cycle channels are present: people comment on tracks, share them or remix them. These cycle channels contain activities that happen all the time, but there is no way of actually mapping them technically. SoundCloud doesn’t keep track of relationships between tracks (for example the original and a remixed track). I think object relationships can add a lot of extra value to the platform, it also fits in well with the idea of “linked data” — I’ll talk about that later.

---

10 Cycle channels are: commenting, sharing and remixing, as described in the chapter “the model”
Based on the model and the cycle channels, there are four types of relationships that could be applied to tracks on SoundCloud:

1. Remixes / Rebounds
2. Use of tracks/samples
3. Replies
4. Related content

1. Remixes & Rebounds

This means that a track is a remix and links back to the original. KORG’s iPad apps do a good job in facilitating remixing by having a very interesting integration with SoundCloud: the apps upload both the track and – optionally – a session file. People can then browse to the track, download the session, play with it and upload it again. Unfortunately, that’s where the cycle ends. It would be great if the remix get’s a reference to the original file (and back). By implementing a “remix” relationship, this link gets preserved. A rebound is similar to a remix, but then by the original author, for example a version 2 of a ‘work in progress’.

2. Sampling

The second type of relationship is “Sampling”. For example, a user records a sample of water and posts it to SoundCloud under a Creative Commons license. After a while, another user finds this sample and uses it in another track. This is an example of how a sample on SoundCloud could be used. The “sample” relationship would allow a track to link back to one or multiple samples that were used to create a track. This type of relationship has a lot of interesting benefits, and I’ll come back to that later in this chapter.

3. Replies

At the moment, a track object on SoundCloud can have text comments related to it. However, it would make sense for SoundCloud to also allow users to reply to a track via audio. With “Takes Questions”, SoundCloud has already explored this type of relationship a bit, but I think audio comments can be taken a lot further by allowing users to reply on other content, maybe even non-sound objects. I will go into that more later.

---

11 I borrowed the term "rebound" from Dribbble:

A rebound is a shot in reply to another shot. Rebounds offer a way to follow up and “link” two shots together. For example, a sketch can be rebounded with a finished design, or alternate concepts of a design can be rebounded so there’s a reference trail to follow.

12 Sample relationships are especially interesting if you look at the flow involved when searching for Creative Commons licensed files on SoundCloud. For example: someone searches for a sound of water flowing, and then uses it in his track. His track then becomes a way for the sample file to be promoted and credited.

13 See the chapter “The world of Objects”.
4. Related content

This type of relationship is about adding context to a new sound on SoundCloud. The most obvious example would be podcasts (linking to discussed topics), there could be lots of things that are related to your track. The application that generated it, or an event, etc. This object relationship could potentially allow people to speak up about *anything on the web*, since a track can be “related” to a website, a video, post, image, movie or anything else. It combines well with my object ecosystem concept. Instead of linking to a url, you could link to a specific object.

**Metadata**

When implemented, it could add four sets of metadata to a track:

1. The files the track was based on (by sampling)
2. The track that was remixed.
3. A reference to a track that the current track is a reply on
4. Related content on the web

Of course, this can also be turned around:

1. Tracks that use “this sample”
2. Remixes of this track
3. Replies on this track
4. Content on SoundCloud that is related to something (a url, object)

When looking at the first three types, SoundCloud would effectively be recording the history of a file as it iterates through the different steps in the model. For a possible way to use these relationships for remixing, see my concept on the remix button.

The fourth type of metadata is very interesting, especially because it can also be used backwards (this would allow audio comments, for example). It allows you to ask the question “What content is related to this URL?”, for example.

**Timing relationships**

The types of metadata that I’ve just described are very useful, but they become even more useful when they could be linked to a specific time. Merely listing the samples that were used in a track is useful, but marking when samples start and end is much more useful — it would make identification of samples easier. The same goes for mixtapes; knowing not only which but also *when* tracks are featured is important. Giving related content a specific time can be useful as well, for example when linking to specific topics in podcasts.
So where’s the added value?

I think object relationships are a significant step forward for SoundCloud, both in user experience as in a technical area. Primarily it gives the tracks on SoundCloud an extra dimension of meaning — “relationships”. To quote Tim Berners-Lee on the subject of Linked Data:

*Data is relationships. This person was born in Berlin, Berlin is Germany. And when it has relationships, whenever it expresses a relationship then the other thing that it’s related to is given one of those names that starts with HTTP. So, I can go ahead and look that thing up. So I look up a person — I can look up then the city where they were born I can look up the region it’s in, and the town it’s in, and the population of it, and so on.*

SoundCloud could be one of the first big web services to actively pursuit the idea of linked data/semantic web.

Relationships also generate insight and a wealth of interesting metrics on how sound is used on the platform: where they originate from, how many tracks get remixed, how samples are used, etc.

By implementing concepts based on the relationships, SoundCloud can stimulate the culture of sampling and remixing on its platform. Track relationships give creators more credit for samples that they have created because they are featured on the page of the track that sampled them. The same goes for remixing, this can increase people’s audience because they are featured on the original track’s page.

Creators are able to get insight into who’s doing what with their content, making it easier to have remix contests, for example. Consuming users discover new remixes of their favourite tracks more easily — you could even go as far as showing new remixes of your favourite tracks on the dashboard. It will also be easier for users to see related content, especially useful for podcast listeners.

It also reinforces the message that SoundCloud deals with all kinds of sounds, a truly open platform for sharing sound. Because it facilitates people in making their content more accessible to other creators, people will grow more attached to the platform.

As I explained, relationship data isn’t just about other tracks — it can also be used with objects on other sites. If something is relevant to a YouTube video, a blog post or a Foursquare venue, for example, it could also be linked. This is also a way of generating traffic to other websites and – if done properly – also generating traffic back to SoundCloud.
The API

By building track relationships into the API, developers could unleash all kinds of new ideas on the platform. Relationships could stimulate the remixing ecosystem on the platform, but the moment that SoundCloud involves app developers and makes a standard for sharing session or sample-kits, then there is something in there for everybody. And I think that it fits SoundCloud to make sound on the internet better for everyone.

Example of relationship API usage

There are a number of reasons why object relationships and similar types of metadata should be made available through the API:

- So music-making applications can download the samples that are used in a track as a starting point.
- People can build cool mashups with this data – for example making a [mashupbreakdown](#)-like app.
- Allowing applications to view related tracks

I think the references should be included in the default dataset that one receives when asking the API for information on a track. There should – however – also be ways of getting specific relationships:

- Getting all the relationship data for a track
- Getting all the remixes for a track
- Getting all the samples used in a track
- Seeing which track this is a remix of
- Seeing which tracks sample this track
- Seeing what content on SoundCloud is related to a specific url/page/etc.

Audio Comments

Audio comments is a concept built upon the idea that one track could be related to another because it is a reply or answer on another. In practice this means that instead of just a textfield, SoundCloud could also display a record button underneath a track to accept comments. Instead of merely typing what they think of a track, users can actually express it by talking about it.

SoundCloud wants to unmute the web, and in order to do that I think the concept of audio comments can be taken a lot further than comments on SoundCloud. Object relationships would allow users to create a link or relationship to other content (that doesn't have to be on SoundCloud). With the relationship in place, other users will be able to see what the track was a

---

14 Not just small developers, but also developers of big audio-workstation app developers like Ableton, Native Instruments and Avid could collaborate with SoundCloud on universal session files.
comment on. In the meantime it will also allow the creator of the original content to see what comments were made on the content.

After uploading a track, users can add relationships to it by typing in URL’s. In this example, the website recognised an earlier input as a Foursquare venue and so it is able to show an image with it.

There are three levels in which SoundCloud could support audio comments: The most basic level of audio comments is allowing people to reply on content on SoundCloud. The next level would allow comments to be a reply on "anything" on the Internet. Instead of linking to a track, the comment could be linked to a URL of a blogpost, website, video, photo, etc. The track’s page will show what the comment is about, but because of the relationship the other applications can use the SoundCloud API to do the reverse: find out what content on SoundCloud is related to their content.

The next step would be for SoundCloud to supply a service that makes it easy for other websites to accept audio comments and display SoundCloud content related to their url. This is similar to what Disqus does: a service that makes it easy for any website to accept comments.
CloudComments is a prototype that I made in order to realise this. By embedding a frame on a website and sending an identifier for the post with it, the system retrieves available comments and displays a record button underneath them. The newly recorded comment is uploaded to SoundCloud and is linked to the original content in the form of machine tags and a textual reference in the title and description of the track. If Object Relationships were to be implemented, the machine tags wouldn’t be necessary as object relationships will be the solution to linking a comment to an article.

The combination of these concepts allow SoundCloud users to speak up about anything, while allowing content owners to see what is being said about their content.

**Remix Button**

In my chapter “Object relationships for SoundCloud” I described a number of relationships between tracks on SoundCloud. One of the relationships was remixes. It allows tracks to be marked as a remix of another track. As described in that chapter, KORG’s iPad apps are currently the only apps to facilitate remixing other user’s content by allowing users to upload the session file as well. The session file is then hosted by SoundCloud and then referenced in the track object. However, when you upload a track that was based on another track, no reference to the original is saved in the original track. Object relationships will enable this reference to be saved.

I was thinking of a way to facilitate remixing of other people’s content with other music applications as well — so not just the KORG apps, but also apps like Reason, Logic, ProTools, Ableton Live, etc. A way SoundCloud could facilitate this is by allowing creators to upload session files and allowing other users to download them through a “remix button”. The resulting remix can then be marked as a remix of the original track.

It is worth mentioning that while the KORG apps’ session files are not too big, session files from other audio applications can get huge. However, the size of the session files is not really an issue, since SoundCloud is a storage platform. They could also collaborate with services like Gobbler — a service that specialises in the storage, backup and sharing of audio session files.

---

15 When storing session files on SoundCloud, it is important to realise that you don’t necessarily need to host the production session file. It could just as well be a session file with renders (“bounces”) of each track or just a “kit” with samples. This would solve some practical problems caused by dependancies of the session file — the creator of the original track might have used (proprietary) plugins to process the tracks that the remixer doesn’t have.
Sample Workflow

Working with object relationships can have a significant effect in the sample community on SoundCloud. It allows for easier crediting of people and therefore gives those people a lot more eyeballs than they would initially get. It also shows other users *what can be done with a specific sample* and how it was used and manipulated in a certain track.

However, simply saying that “track A used sample B” is easy to do, and although it will improve the eyeballs that a sample and its creator get, it doesn’t necessarily have a lot of added value for other users. A way of adding more value to the experience of users visiting a track page is by giving more insight into exactly how a sample is used — e.g, showing the occurrences of a track. A sample starts at a certain time, might loop a while and then ends at a certain time. Storing this data together with the reference to the sample could provide that added value for both users and developers using the API.

Examples

The most obvious way of getting this data to the users is marking samples in the SoundCloud player. Getting this data available to users can have a lot of added value for listeners of DJ mixes, because listeners would be able to identify the track that’s currently playing in the mix or see a list of all the tracks in the mix. Now since not all tracks contain samples that are exclusive to SoundCloud, it would not be very useful if the relationship system only supported objects on SoundCloud. It could also refer to a URL of another object.

For developers using the API, using this time data can lead to a lot of interesting mashups. For example, visualisations of the different samples in a style that’s similar to the website *[Mashup Breakdown](#)*.

Getting the actual data

Before making the references on SoundCloud, there must be a way of getting the data (the occurrences of a sample in a track). One could manually input this data, but I think that the best source of this data would be the audio applications that the track was exported from.
How to track sample usage in songs

Until there is a cross-platform way of storing metadata in a file\textsuperscript{16}, there needs to be another way of registering the source of a sample that is used. One way of doing that would be to load the sample from within audio applications. KORG’s iPad apps already allow you to download other users’ creations and build upon them, and this is the way that I imagine it would work with other audio applications as well.

A way to realise this is to load samples from SoundCloud from a sample browser inside an audio application. This way, when importing a sample using the sample browser the audio application knows the source (url) of the sample that is used.

Now that the audio application is aware of the source and other metadata that belongs to the sample, there needs to be a way to save this data along with the track when it’s exported. Again — because there is no standard way of saving this data inside a file, it needs to be saved separately. There are two possible ways of doing this:

1. **Saving a file along with the track** that contains a list of the samples that were used, where they came from and optionally other data like the timestamps that they were used. *This data could also be automatically uploaded with a track if an audio application supported a function that exports the current session to SoundCloud*. SoundCloud can then display the samples that were used. It can also include basic metadata (bpm, key, title) so the user doesn’t have to fill in that kind of information himself.

2. **A universal format for sharing sessions from audio applications.** Ideally, every audio application would save its session file in the same way. Unfortunately this is not the case, most importantly because not all audio applications work in the same way. Instead of suggesting that there should be a standard format for audio sessions, it is a good idea of having a universal format for *sharing* a session between audio applications. The idea is that a user could choose to export a session file to this format. It could include the following types of data:

\textsuperscript{16} There’s no real cross platform way to store metadata with files. The Mac OSX filesystem does support metadata but it is not usually used for this kind of metadata. However, Apple’s Safari and Google Chrome do utilise this by saving the online URL of the file in a “where from” field, accessible when using the inspector on a downloaded file.
1. The individual samples that were used in the session — including their metadata and occurrences

2. Track/plugin settings and automation

3. Source files or renders of the samples used

4. Basic track metadata: beats per minute, title, etc...

Although this format is pretty simplified, it is enough to transfer the most important information about a session so that it can be shared between apps or users. When uploading this session file to SoundCloud it is possible to display the different occurrences of samples. It also allows users to facilitate remixing of their own tracks by displaying a remix button next to the track. This button would then download the session file so that the user can import it into his favourite audio application.

The idea of this universal session file goes beyond SoundCloud, of course — it works in all situations where someone would want to share an audio session. That's why it is important that the format becomes a standard. SoundCloud has a good reason to push it because it can increase the overall usage of the platform, but I don’t expect SoundCloud to develop the standard all by itself. The best option would be to collaborate with a developer of audio applications on the standard and have one app natively support the standard, then making the standard public. Hopefully, enthusiasts will then pick up the standard and create solutions to make it work with other apps.

How To Implement Object Relationships

In the previous chapters of my thesis I’ve defined a number of different object relationships and created a number of concepts based on those relationships. This chapter describes a roadmap of implementing object relationships on SoundCloud.

1. Defining relationships

For object relationships to work out it is important to define a good basic set of relationships. The subchapter “Types of object relationships” defines a number of relationships that can be used. However, relationships could be more than linking tracks together, it could also be about linking tracks to related content somewhere else on the internet.

2. Preparing the site for relationships

The next step is preparing the site for the different object relationships. This includes preparing the database for relationships to different types of content and designing a good user experience.

---

17 Universal session files could show the places (in time) where the sample is used, start/end times, how many loops, etc…

18 The problem with transferring sessions from one computer to the other is that computers doesn’t always have the same plugins. This also even an issue between different software applications. Still, the same automation could then be mapped to on another – similar – plugin
There are two big challenges on the design front: making it very easy to add relationships to a track and making the idea of relationships clear to the end user — “relationships” might not even be the best phrase to use. Audio comments are a way of making the idea of object relationships clear. Another option would be to sending remixes of favourite tracks to the user’s dashboard.

In my opinion, the best place to create a new track with a relationship is the page of the content that you want to relate your new track to. In other words: replying to a track should be possible from a track's page.

This concept shows a number of object relationships in place, first there are audio comments. New comments can be recorded on the same page. This track features the sample "Birds in the forest" and lists it in the sidebar. Because this is a track that can be remixed, it shows tracks that have a remix relationship. The artist also offers a sample pack that allows people to remix the track.

When replying to content on another website, there could be a bookmarklet, browser extension or a button that would allow you to record something and create a relationship to the content. This would allow users to speak up on anything and share it to their followers with the context of the relationship. This way, the idea of object relationships becomes almost natural.
3. Expanding on the idea

When the site is prepared for supporting the object relationships, the next step is making sure that not only users but also developers can do awesome stuff with it. In other words, SoundCloud needs to facilitate and aid third parties in defining object relationships as well. The first step is making relationships accessible in the API.

A relationship API would also allow developers to search for items that are related to their own content. This could allow them to see what content on SoundCloud is related to their content. This could be used to show a list of comments underneath an article. SoundCloud could also facilitate this by offering a comment service similar to Disqus or CloudComments.

Object relations improve the visibility and discoverability of samples within tracks. In the subchapter “sample flow” I described the idea of a SoundCloud-powered sample browser inside audio applications. In order to facilitate the sample workflow as described in that chapter, SoundCloud could create an API wrapper for a sample browser similar to what’s available in the KORG apps. SoundCloud could also write down some recommendations in handling samples from the platform so the uses of samples can be tracked and saved using SoundCloud’s object relationships.

By making relationships very accessible to the users, SoundCloud can also facilitate the sampling and remixing cultures. By using a universal format for sharing sessions SoundCloud can facilitate creators when they want to have their tracks remixed by other users. Big artists could be approached to be the example in this case.
Case: Business aspects

In the thesis I have described multiple concepts that have an effect on SoundCloud. This chapter contains a more detailed look at the different business aspects of these concepts.

SWOT Analysis

After my own analysis and an interview I created a SWOT analysis that summarises SoundCloud’s current position and opportunities.

**Strengths:**
SoundCloud is very strong as a platform, with a large community that has a big affinity with it. It’s a strong product that’s still growing and has a powerful infrastructure to support it. SoundCloud also has a very clear mission to be the best platform to store any kind of audio on.

**Weaknesses:**
SoundCloud is an all-purpose audio platform which means that it does not tailor itself to niches, audioblogging for example. They focus primarily on creators, which is why there are not a lot of consumption-related features (e.g., better content discovery). There is also a generic lack of proper metadata. A lot of metadata is absent or incorrect, but there is not always space for more relevant metadata.

**Opportunities:**
SoundCloud wants to prevent feature creep on the main website, but can tailor smaller needs (and niches) through SoundCloud Labs projects. This is an interesting concept that could mean a lot for the platform. They also have a lot of partners which can — for example — develop applications for niches. Other opportunities are better integration with audio applications. There is a reasonable remixing culture on SoundCloud, but this could be stimulated more. One big opportunity is audio comments. It’s a way of unmuting the web by letting people speak up on content on SoundCloud, but also other places.

**Threats:**
Although the effect of this is small, copyright conflicts can make some users stop to use the platform. Also, because of the strong growth of the platform there is the threat of scaling issues. Another threat is specialised services: services that tailor to a niche and do “one thing well,” for example. If a company that has more resources and is bigger enters the same space as SoundCloud, then that company could also become a threat.

Business Aspects Of My Concepts

Audio comments on anything

I think audio comments would be the easiest way to introduce the concept of object relationships to users of the website. However, audio comments on tracks are just a small part of the big picture
that I’m trying to convey in this thesis. SoundCloud could offer a set of tools that allow people to speak up about anything. Tools like browser extensions, buttons similar to Facebook’s share button, hosted comment forms, etc. When implemented on other websites, this increases the presence of SoundCloud on the internet and once again reinforces the message that SoundCloud isn’t just about music. It helps unmute the web. In the meantime, the websites that are on the receiving end of this get richer comments. Audio comments are also an additional touchpoint for SoundCloud as it can be seen on more kinds of websites. Visitors of websites that normally wouldn’t have anything to do with SoundCloud, can now get exposed to the product through the audio comments forms.

**Using samples**

When using object relationships to refer to samples that were used in a track, creators get more credit for samples that they created because they are featured on the page of the track that sampled them. For SoundCloud this generates insight into how tracks are used. This is also a good moment to discuss possibilities with application developers. For example, putting a SoundCloud sample browser inside applications like Ableton Live is a way of making people more comfortable or familiar with the platform whilst – again – reinforcing the message that SoundCloud is also a great tool for finding samples.

**Remixing**

The remix relationship could be an important tool in stimulating a culture of remixing and reusing content. It also facilitates users in sharing their content and allows consuming users to discover new content related to the content that they like.

At the moment there is no real way of being sure how much people remix each other’s content, since this is not mapped. However, statistics show that 5.5% of the tracks contain the word remix in the track title and 8.44% of the tracks contain that word in either the title or description of the track. These statistics do not necessarily include the usage of other people’s samples and could also be remixes of content outside of SoundCloud. Still, they do give a reasonable insight in how much content on SoundCloud is a remix. In order to get better statistics on this, SoundCloud could start to support a "remix" relationship.

There could be multiple levels of supporting this relationship: from simply supporting the type of relationship to actively facilitating the remixing of content. Like other relationships, it improves the insight that SoundCloud has in how content is used. At the moment, stimulating this actively is not a real priority for SoundCloud, but supporting it partially might facilitate users in remixing more and at least gives SoundCloud better insight in how much users remix content, so that decisions can be made on that data.

**Universal session format**

The KORG apps make it very easy for people to remix or re-use your track, because the app allows you to upload your track as well as your session file on SoundCloud. A universal session file
would do the same for other audio applications. However, this standard doesn’t exist yet. SoundCloud could collaborate with the developers of audio applications to make this a reality.

In the end, for SoundCloud this means that it can also promote itself as a platform for storing session files (e.g., for collaborations). Because universal session files contain both metadata and references to samples, more insight can be received in how content is used: where tracks originate from, how many tracks get remixed, how samples are used, etc. Also, because not only users but also the applications that created the session files get attributed, the discoverability for apps increases.

For users, universal sessions give better portability between apps. Getting sessions from basic to more advanced apps, for example. Third parties will get more exposure and thus are more discoverable, but next to that there are no real benefits for them to support this.

Related content
Relationship data isn’t just about other tracks — it can also be used with objects on other sites. If something is relevant to a YouTube video, a blog post or a foursquare venue, for example, the URLs to those objects can also be linked. This is also a way of generating traffic to other websites and — if done properly — also generating traffic back to SoundCloud.

Object Relationships in general
I think object relationships are a significant step forward for SoundCloud, both in user experience as in a technical area. Primarily it gives the tracks on SoundCloud an extra dimension of meaning — “relationships”. Relationships also generate an insight and a wealth of interesting metrics on how sound is used on the platform: where they originate from, how many tracks get remixed, how samples are used, etc.

By implementing concepts based on the relationships, SoundCloud can stimulate the culture of sampling and remixing on its platform.

By building the functionality into the API, developers could unleash all kinds of new ideas on the platform. Relationships could stimulate the remixing ecosystem on the platform, but the moment that SoundCloud involves app developers and makes a standard for sharing session or sample-kits, then there is something in there for everybody. And I think that it fits SoundCloud to make sound on the internet better for everyone.

Object ecosystem
The object ecosystem concept would make sure that user profiles are complete right from the start: Names, avatars, social networks, email addresses and — depending on the solution used — maybe even friends, top artists, etc. Depending on the solution other data might also be available. The data would also be more recent and possibly more complete since the user maintains it for all kinds of services and syncing would make sure SoundCloud receives the updated info.
automatically. This allows SoundCloud to better tailor its service to users, for example: it can recommend friends or artists to follow.

**Persistent metadata**

The idea of persistent metadata is that comments and similar data would be synchronised across different occurrences. In other words: comments made on a track on SoundCloud will be available to sites that display the widget (for example), and the comments made on that page will then be visible on SoundCloud. Again, this supports primarily the users. They will be able to follow conversations happening around a track, increasing the amount of feedback that a creator gets on his track.

However, this metadata doesn’t have to be comments. It could also be about other data as well. SoundCloud could profit from other websites tagging content. A simple example of this would be a bird-spotting club that tags outdoor recordings with birds that are audible in the clip. This data is then valuable for both that website and SoundCloud, as people might be searching for sounds of specific birds. Other websites can then also befit from the combined efforts of the different parties.
Conclusions and Opportunities

This chapter contains the most important conclusions of my research and the opportunities that I see for SoundCloud in the future.

In the analysis a pattern can be seen in how objects are used and how they are being interacted with. They all go through a similar process. They get created, shared, published and consumed. Users can then add value after this consumption, mostly in the form of comments but also in the form of re-using or remixing other people’s content. The model for object interaction maps these phases on a cycle. It displays a common process that objects go through. It can be used to evaluate an application that uses objects. The different steps and “cycle channels” (remixing, sharing, contributing) can be renamed to what’s appropriate for an application after which opportunities can be discovered for increasing the user experience with those processes.

Sound is an interesting object, it is much richer than text but also more accessible than video. It can also make use of all the cycle channels. The model shows how new objects can be created through the cycle channels that are defined in the model. For example, a comment or a remix by a user is a new object. That object is then related to the source object.

Remixing as a relationship is already used on the website, but they are not explicit. Analysis of track metadata shows that between 5-8.5% of the tracks on SoundCloud are probably remixes (See Case: Business Aspects), and I think there is a potential in explicitly connecting tracks together that either remix or sample other SoundCloud tracks. This allows SoundCloud to get better insight in how content is used and allows users to discover new content related to their favourite tracks. It might also stimulate the mixing culture and increase content discoverability for users.

Another cycle channel from the model that translates well to an object relationship is commenting. At the moment, comments on SoundCloud belong to a track, but these comments are only text. Allowing tracks to be marked as a reply on another track (or anything else on the internet) is what would make it possible for SoundCloud to support audio comments.

It is a good way of introducing users to the concept of track relationships. Audio Comments are a huge step forward in unmuting the web and there aren’t any companies exploring this space. I think SoundCloud could enter it relatively easy because it already has the infrastructure to support it. It is also another chance to reinforce the message that SoundCloud is not just about music but all kinds of sounds.

Future

Relationships are the key to the concepts in my thesis. Supporting relationships would allow the addition of more meaning to tracks. It also allows SoundCloud to map how sounds are used and
where they come from, giving it more relevant statistics and metrics that can be used for improving the platform and its experience.

Replies would open the way for audio comments and SoundCloud could lead the way in unmuting the web by supporting this on their website or allowing users to speak up about anything using things like browser extensions, buttons, etc.

I would recommend SoundCloud would collaborate with more app makers (also those of big audio apps) to get SoundCloud inside music making apps in the form of an uploader and sample browser. That way more users will be confronted with SoundCloud but it will also allow SoundCloud to receive extra metadata about the track that is uploaded, data that the user doesn’t have to enter on the site.

Another way of getting this data would be to collaborate with app makers to create a universal session format. A way for audio applications to open sessions made by other applications and for platforms like SoundCloud to get more data which can be used to improve metadata or user experience.

The Answer To The Research Question

In the introduction of the thesis I posed the research question:

“How can sound data exchange between an online application and audio distribution platform and applications improve the user experience and create new business opportunities”.

With support for object relationships the user experience will be improved for people that create samples, or re-use other content on SoundCloud because their content will be more discoverable when the relationships are visible. Consuming users will also profit from this as they can find content that is related to a track that they are listening to. If they like it, they might want to listen to a remix, for example. Audio comments can add a whole new dimension to the web.

Integration with audio applications will allow people to easily find new content for their tracks but it also serves as a promotion for the platform. SoundCloud-based Sample browsers and easy uploading from within audio applications could add a lot of value for users, while in the meantime getting access to more data.


Appendix: The Model

Create → Instance → Publish → Meaning / Value / Context → Consumption

- A user creates an object
- The object is made into an instance or digital representation (e.g., a file or widget)
- The instance is published on a platform or website
- This platform or website displays the object in a specific context. This context gives (new) meaning to an object and adds value through adding metadata.
- The object is consumed (read, viewed, interpreted, listened to, etc.)
Appendix: Wireframes

These wireframes show a number of scenario’s related to the concepts that I described in my thesis: the object ecosystem, persistent meta and object relationships.

Setting up

After a user connects an object server (or a router) to the network, the operating system could ask the user which information he would like to store on the object server. This includes information like contacts, but also data like a music library info or data produced by applications (Last.fm charts in this picture). The same could happen on smartphones. Instead of adding a Google Mail or Exchange account, a user can instruct the phone to use the object server for storage.
When registering on a website via an object server, the website might ask permission to get access to a list of resources on the object server. The browser then shows a dialog in which the user can authorise the website in accessing the resource. The user can see which resources the application wants access to, so he can make an informed decision.
After the user has authorised an application to use data from resources, a lot of profile data can already be visible on the website. This allows applications to improve the experience for the users, for example by give better follow recommendations. The wireframe on the right shows the “find friends” tab that is present on SoundCloud. In this case it uses data not only from Facebook, but also from object servers. SoundCloud found five users in the user’s address book that have SoundCloud, but also found 8 artists in a “top artists” resource that can be followed on SoundCloud.
After a user creates a new account on Tumblr, the site could ask permission to write data to the “website lists” resource. This would allow it to add the newly created blog to a list of websites and social networks that the user has.
Persistent Metadata

This wireframe shows persistent metadata at work: The music blog embeds a track from SoundCloud and displays comments underneath the article. Some of the comments come from sites other than this music blog, in this case SoundCloud and other music blogs. This is because the comments are added to the track object on SoundCloud in the form of a standardised “comment” object. The object ecosystem then makes sure this data is synchronised between different websites.
The user records a sound and uses the autosharing function to post the new track to Tumblr. The wireframe on the right shows the track page on SoundCloud. Because of the object ecosystem, the track page can now display where the comment is used/embedded and how many likes it has.
This concept shows a number of object relationships in place. One of them is audio comments. In this case, a record button was added to the comment section so people can comment on tracks. The second type of object relationship in this wireframe is samples. This track features the sample “Birds in the forest” and lists it in the sidebar. The owner of this track has uploaded a sample pack for this track, so a “remix this track” section is added in the sidebar. To promote tracks and artists that have remixed this track, a list of remixes is also added to the sidebar.
Object relationships can contain timing information on SoundCloud, this allows users to see when a sample was used. This is a concept of a SoundCloud player widget that visualises this information. The idea behind this concept is that a standard widget can be expanded to show this information if it is available.

After uploading a track, users can add relationships to it by typing in URL’s. In this example, the website recognised an earlier input as a Foursquare venue and so it is able to show an image with it.