# *Protagonist* Feasibility Study

Prepared for: Katie Day, The Other Way Works

Prepared by: Jason DaPonte, Managing Director

21 May 2014

This document is a response to The Other Way Works *Protagonist* Requirements Document that assesses the feasibility of delivering the requirements. Where the requirements are deemed to be feasible, we have suggested an approach. Where they are deemed difficult or infeasible, we have suggested alternative approaches.

We have followed the structure of the Requirements document to ensure that the documents build on each other. The requirements are listed in the left-hand column in the order we have addressed them (rather than the numeric order they are presented in the original document).

Table of Contents:

# Overview

We have reviewed a number of third-party solutions for providing various parts of the solution and are recommending a largely custom solution that tightly fits the requirements.

There were no third-party solutions that can provide the entire experience as outlined in the requirements.

Solutions were found for various components in the system, however, given the nature of the project and the budget, we feel that a custom solution is likely to be more cost effective and less risky. This is for two reasons: scalability and integration risk/cost.

Scalability is an issue because in our understanding of the requirements is that this is for a relatively small (hundreds) of users at any given time. The commercial video solutions (serving platforms, players, etc) are generally created for massively high use (millions at a time) and so will be over-specified for the needs of this project and would therefore be overly costly for this type of project – particularly at prototype phase.

Integration with the systems we evaluated also caused risk and cost issues. Doing an integration like this would require piecing together a number of existing solutions to meet requirements they weren't really designed for and then connecting them with custom software acting as 'glue' between the systems. In our experience this is only successful when the systems are designed to meet the same requirements. In the case of *Protagonist* small pieces of 'middle ware' – software that connects systems – would be required.

Overview of feasibility of top-level requirements (elaborated on in detail later in the document). Where the items are classified as 'Difficult' or 'Infeasible' this is based on the feasibility of delivering the requirement *as written*. It is not to imply that the project is impossible and we have suggested alternative approaches for most of the items.

1.0 Harvest Social Media Content – Feasible
2.0 Categories social media content – Largely Infeasible
3.0 Identify significant content – Feasible
4.0 Select content – Difficult/Infeasible
5.0 Select content from predefined dates – Feasible
6.0 Put content on film timeline – Feasible/Difficult
7.0 Master video – Feasible

## Major Issue: Speed

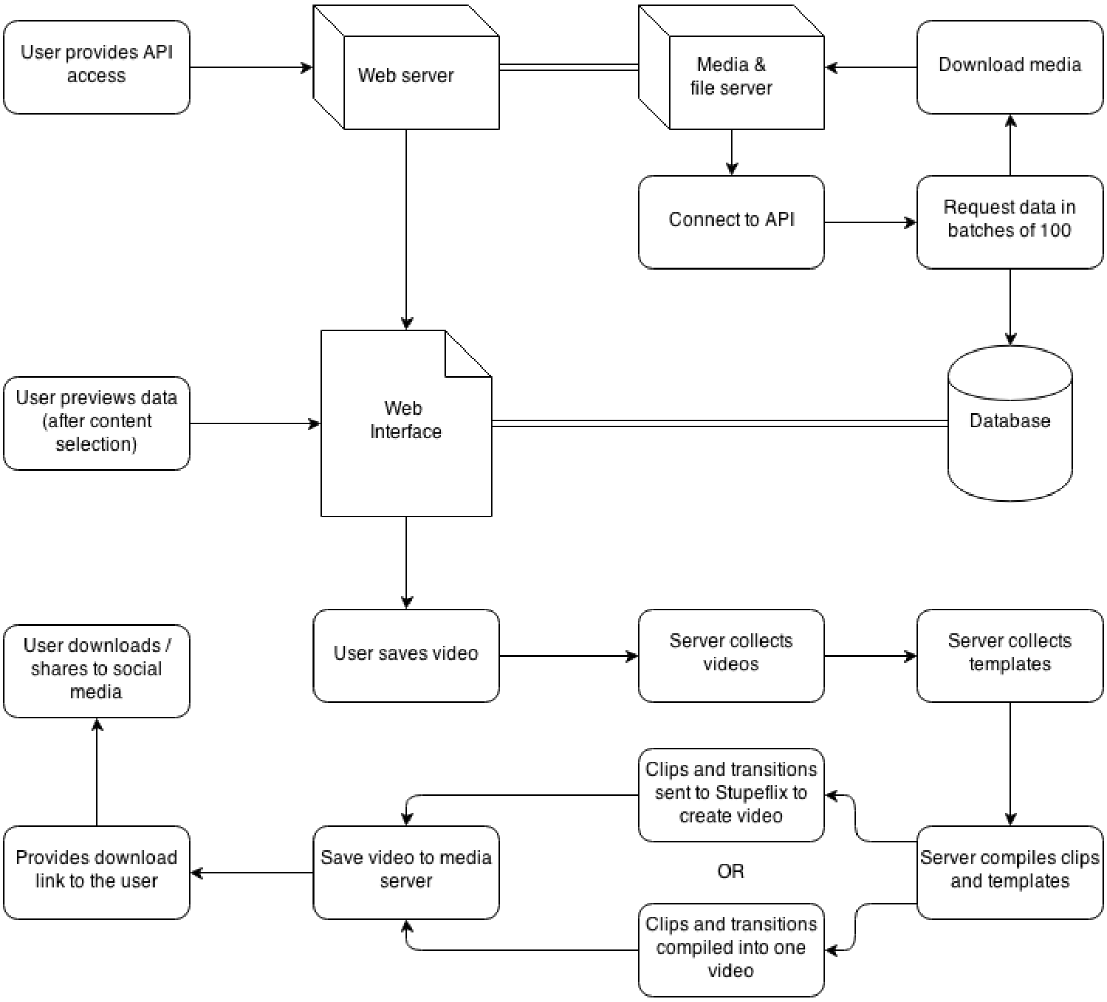
The proposed application involves mining, storing and processing tremendous amounts of data for each user. It also involves connecting to a variety of sources to get the data, which will take time. It's very difficult to predict, without actual prototyping and testing, how long this will take. Even with testing, variables like speed of response from the API servers and the amount of bandwidth available will create large variables.

While much of the solution can be achieved through automation, a level of waiting will be inherent to the experience. We would strongly recommend building this waiting time into the overall experience architecture (for the digital and performance elements) to avoid creating problems in the performance space. We would be happy to brainstorm/devise this with The Other Way Works if we continue to work together. We could also carry out testing to try to create a better time estimate and then make better recommendations for where the slow-downs are and how to potentially avoid them.

Most APIs have a maximum number of queries per day – usually in the thousands, which could also limit access if the application became publicly available and very popular.

# Workflow Architecture

The following diagram has been created to outline the workflow and major functions of the system as we propose creating it.


User provides API access
Web server
Media & file server
Download media
Connect to API
Request data in batches of 100
User previews data (after content selection)
Web Interface
Database
User downloads / shares to social media
User saves video
Server collects videos
Server collects templates
Clips and transitions sent to Stupeflix to create video
OR
Clips and transitions compiled into one video
Provides download link to the user
Save video to media server
Server compiles clips and templates

# Technology

This section outlines the technology we would recommend for the prototype and some of the related issues.  It is aligned with the Workflow Diagram above.

## Servers

Media Server:

This will be a powerful server containing all downloaded files and the database. It will run the automated data ingestion, as well as video transcoding and final compilation of video.

Media Server Specification:

Linux (Ubuntu or CentOS) with min. 24GB RAM. HD at least 2TB.

Web server:

This will run the website (built with PHP and HTML5) and will have a live connection to the media server database and files.  It will require an SSL certificate.  This is the server that the users will connect with and use to both connect their social networks, and preview and select the content.

Web Server Specification:

Linux (Ubuntu or CentOS), 8GB-16GB of RAM.  This must be a dedicated machine; not a virtual server or similar.

## Connectivity and Hosting

Hosting for both servers should be purchased with redundancy and 24/7 support and guaranteed uptime that is deemed suitable by the management for the project.

Connectivity and file storage is very important. These servers need a lot of bandwidth and hard drive space – though suggested figures would be dependent on the expected number of concurrent and total users, which has not been specified in the requirements.

## Video Tools and Formats

We would use native HTML5 video players. This enables the fastest playback on the most devices, as well as fine-grained controls over seeking, playlists etc in a consistent way.

We would work in HTML5 format, which means predominantly H264 format. We will need to transcode video clips where necessary (where these formats are not provided by the social network’s API) into WebM and OGV format to make sure we cover all browsers.

## Workflow and Feasibility In Detail

This section goes into detail about how feasible each of the top-level requirements is and how we would recommend building them (or not).

| | |
|---|---|
| 1.0 Harvest Social Media Content | Status: Feasible.<br>Approach:<br>Before social media can be harvested, users will need to sign in to their social media accounts to give permission for their material to be accessed, and this may be one of the most complicated parts of the experience from the user perspective.<br>This is done in different ways for different networks:<br>● Facebook Connect[1]<br>● Twitter OAuth[2]<br>● Flickr OAuth[3]<br>● Instagram OAuth[4]<br>● Youtube OAuth[5]<br>● Vimeo OAuth[6]<br>We will set up necessary apps for Facebook and Twitter, which will let the users click a button to log in with those services and grant us access. For the OAuth APIs, it is a similar process whereby the user logs in to the service and confirms that our website should be able to read data on the user's behalf.<br>This process will be made as easy as possible, and the user should connect their social networks at least a week before processing a video, as it will take time to ingest all the data. An email can be sent to the user when data ingestion from all connected services is complete, or this can be presented to them as part of the performance or on entrance to the performance environment.<br>Other networks (as specified in the Requirements document) could be added but with manual login processes.  Doing this introduces an element of risk because any change on their site/service could break the |

[1] https://developers.facebook.com/products/login/

[2] https://dev.twitter.com/docs/api/1.1/post/oauth2/token

[3] https://www.flickr.com/services/api/auth.oauth.html

[4] http://instagram.com/developer/authentication/

[5] https://developers.google.com/youtube/2.0/developers_guide_protocol_oauth2

[6] https://developer.vimeo.com/apis/advanced

| | |
|---|---|
| | login systems (whereas with Facebook Connect and OAuth, provisions are made to accommodate legacy login interfaces). |
| 2.0 Categorise social media content | Status: Infeasible<br>The Requirements document references categorizing significant events in the users life and organising these into themes.<br>While there is technology that tries to do this, it is not very reliable and is subject to edge cases that would make the system appear ‘broken’ or ‘wrong’ to the user.  Also, the nature of someone’s life may not be expressed through these traditional ‘moments’ – for a musician, their top 10 concerts may nearly tell the story of their life, for example.<br>The only reliable ones that we would recommend using (which some users may not have entered) are from Facebook and are:<br>- Birthday<br>- Life Events<br>- Works at/studies at<br>However, even with these, it is worth noting that these may still not be accurate as some people use them in a joking way (eg. “Works at: School of Hard Knocks”) that could dilute the output of *Protagonist*.<br>We would strongly recommend that using these types of facets in the data not be designed as part of the ‘promise’ of the product but rather be considered ‘added extras’ if they are incorporated and work correctly.<br>We have suggested two alternative approaches for creating a user interface that helps elicit this type of data from the user later in this document in the “Preview / Manua Ranking Process” section and believe that this is a more workable solution that fits with the theme of the project and that won’t appear to be ‘broken’ to users. |
| 3.0 Identify significant content | Status: Feasible.<br>Approach: |

This would be done through a data ingest process and an automatic data ranking process (a ‘preview and manual ranking process’ will fulfil other requirements later in this document).

Data Ingest:

The first part of meeting requirements 2.0 and 3.0 is the data ingest process.

There will be an automated process (‘cron job’) on the dedicated server that will automatically download as much data as possible from each user’s connected accounts.

This data will be split into as much metadata as possible depending on the source:

- Facebook Status text, tagged friends, # likes/shares, comments, attached media, geo-location, date/time
- Twitter status, location, hashtags, retweets, replies, favourites, attached media, date/time
- Flickr media, caption, location, tags, geo-location, camera info, date/time
- Instagram media, caption, comments, hashtags, tagged users, geo-location
- Youtube media, duration, comments, tags, date/time
- Vimeo media, duration, comments, tags, date/time

This data will be stored in the MySQL database and the user’s database record will keep track of how much data has been imported, and which services have completed the data ingestion.

Media will be stored on disk and a reference to the local copy and the online copy will be stored in the database.

Automatic Data Ranking:

The second process required to meet requirements 2.0 and 3.0 is the Automatic Data Ranking process.

‘Inherent Ranking’ will be created applied based on certain criteria that can be determined wholly by the information as it is received from the APIs.  This is what will allow information from different social networks to be ‘baselined’ and joined.

All of the posts will be added to a user ‘timeline’ as part of their database record and then scored for importance.

| | |
|---|---|
| 5.0 Select content from predefined dates | The elements of this will be based on are below:<br>● Date<br>Date and timestamp will be the common field that unites all of the posts into one XML timeline in the database.<br>- Can be cross-referenced with publicly available calendars for public holidays.<br>- Can be cross-referenced with important dates[7] (such as the Royal Wedding, current affairs moments)<br>- Can be cross referenced with information on the user's social media profiles to look for personal dates (anniversaries, birthdays)<br>Posts relating to significant dates will be upweighted in the algorithm and text for these dates can be added to the videos, from the calendar sources, if required.<br>We will allow manual input for dates to focus on. We can provide suggestions based on the data – i.e. the periods with the densest posting and interaction, as well as notable events of international significance (Wolfram Alpha).<br>We should allow for date ranges to be selected rather than a single day. For example, sporting events that go on for a week, or weddings that include the before and after photos and congratulatory messages. We can make this 'fuzzy' and look for content that has been posted in the same place but perhaps two days earlier than the main date.<br>Once each post is on the 'timeline' it will be scored for importance based on the following:<br>● Post type<br>By looking at media type and length, we can make some assumptions about how important the post was to the user themselves,  based on how difficult it was for the user to create (which denotes intention).<br>Points would be applied to every post, based on looking at the metrics below to help us determine a score for |

[7] http://www.wolframalpha.com/input/?i=Notable+events+in+2013

each post, across all of the content ingested from all of the networks the user has given us access to.

Media Type Score (points for one of the following):

We would recommend creating a hierarchical ranking to score these, along these lines:
- Video (user uploaded)
- Photograph (user uploaded)
- Tweets or status updates with media
- Tweets or status updates (without media)
- A ‘tagged in’ video
- A ‘tagged in’ photograph
- A ‘tagged in’ tweet or status update

Social activity:
By looking at the social activity around each post as a second factor, we can gauge its importance to the community of peers around the user.

Social activity score (sum of numbers below):
- Number of replies and comments (with upweighting factors for: activity in first 24 hours and activity from family members)
- Number of likes, re-tweets, shares (with upweighting factors for: activity in first 24 hours and activity from family members)

The overall score for each item would then be:

Importance = Media Type Score + Social Activity Score

This formula is based on up-weighting video and visual media to help fit with the brief for a ‘filmic’ experience and to identify places where great social interaction has occurred, which generally indicates important moments.

The exact values for the points should be determined during testing of the prototype to create the appropriate creative effect. A future phase of the project could allow fo users to create the scoring themselves based on what’s important to them, or how they perceive their own social media based on a user-friendly questionnaire.

Numeric ‘points’ would be assigned for each of these to every post and then posts with the highest scores would be ‘nominated’ as potential material to use in the final video.

| | |
|---|---|
| | The next process allows users to make adjustments to this ranking. |
| 4.0 Select content | Status: Difficult / Infeasible<br>Meeting this requirement, as written, completely through automation is not possible.<br>However, there are two options that could still produce wha we believe will be a good effect.<br>The first, which is wholly automated, is to rely simply on a selection of the top 10 – 12 posts based on score in based on increasing date, or with simple date rules attached (eg. No more than 1 per year plus the items on the curator-selected dates). The upside of this is that it creates an automated experience but it doesn't ensure that there will be a variety of 'types' of moments over the spread of the person's life.<br>The second, is less automated, and relies on the user making choices about their own posts. Two methods for doing this are described below but while the user scenarios are very different, in either case, the technical approach for this would be to create controls in the interface that let the user have some manual control over some aspects of the ranking algorithm. This is the Preview/Manual Ranking Process.<br>Depending on the creative effect/experienced desired, there are three ways the significant posts could be selected:<br>The first could be done by issuing the user with a series of provocations/questions (eg. "Which of these is your favourite work memory?" to determine work moments or something more cryptic like "What was your happiest age?").<br>The second approach would be to show the user their top 50 weighted posts and let them manually select 10 – 12 to stay on the timeline. Their choices would either select or delete particular elements in their timeline until there were 10 – 12 left. |

| | |
|---|---|
| | Once a chronological view of all data supplied by the user is created, we could also provide controls for the following (if desired):<br>● Start, end of timeline<br>● Duration of output<br>As these are not in the requirements, we have not expanded on them here.<br>Addressing the desire to potentially allow editing of the timeline in future versions after the prototype, we could let the viewer preview in a timeline view, that looks similar to a bar chart; higher parts of the graph show a denser portion of media, which the user can explore by clicking on it to see what media has been brought into each area. Applying higher rank to specified locations, dates, people, keywords (e.g. “Show me more of my birthdays”) would also be possible. |
| 6.0 Put content on film timeline | Status: Feasible / Difficult<br>Output Process<br><u>Length</u><br>The duration of an event in video format will depend on the amount of data available. If there is little data, we can show photographs for longer, or increase the time of the transitions; conversely if there is too much data, we will need to reduce or remove the number of posts included.<br>This aspect may need manual intervention (or this could be offered as a feature in future releases after the prototype), such as allowing the user to define which elements to keep and which to discard, and potentially selecting the start/stop range of video. In essence we are providing rushes selected by the algorithm on a proposed storyboard, and the interactive controls to edit it. This can be in the guise of a full editing toolkit including tools to crop images, rearrange items on the timeline and seek videos, or it could be a set of sliders that control things like ‘Show more photos’ or ‘Show less tweets’ or ‘Show more content that has more likes and comments’.<br><u>Effects, Transitions and Visual Themes</u><br>We can pan and zoom, fade in/out, apply colour tints. We will need to see transitions created by the animator before confirming whether they can be recreated in HTML5 for |

previewing purposes. Transitions can be created and applied using HTML5, depending on the exact specification of what is required.

The animated effects could not be easily re-created using HTML5 video but, this could be achieved with a custom Linux server running open source software. In this case, we can use green screening in the animation in such a way as to place video into parts of an animation built specifically for this purpose.

Templates, transitions and effects should be created in conjunction between the animator and programmer together when the system is being made to achieve the best effect.

Based on the sample video, we have the following observations:

- Content rotating, multiplying and moving: this is entirely possible.
- ‘Cutting up’ and reassembling pieces of a photo: this is probably feasible as long as the shapes to be cut are predetermined between the animator and developer.
- Content moving in and out of envelopes: this is probably feasible as long as the social media content remains on a 2-dimensional plane (eg. It is not folded/unfolded itself)
- Content being folded/unfolded: this is very difficult/potentially impossible because of the 3D/lighting effects required.

Alternatively, we could use Stupeflix (stupeflix.com) for creating the effects and applying visual themes. This provides an API for creating video from mixed sources. We may need to create images out of tweets if we wanted specific textual layouts, but other than that, we can provide images and videos, music and even maps directly to the API for processing with in-build transitions and effects.

The downside of Stupeflix is that API processing is priced per output duration on each clip. It would be worth creating a series of videos and then disassembling them to see how much it would have cost to build on the Stupeflix platform. Accounts are billed at $30 minimum per month.

| | |
|---|---|
| | We considered a number of other options for this step, but discounted them.  They are further described in Appendix A.<br>Templates can be integrated as part of the build process. Since we will be recreating effects, we can control colour, font, and text and image sizes. A template would include a colour palette and a series of transitions. It may also include a preferred duration of image display and speed of transition to further create atmosphere. |
| 7.0 Master Film | Status: Feasible for video; Difficult for Commercial Music<br>Music<br>Adding music would be part of the Output Process. It will need to fade in and out and controls for this can be created We can offer the user a set of predefined ‘generic’ background music tracks that have been rights cleared by the project (eg ‘happy’ ‘sad’ or ‘dramatic’ themes).<br>However, using streaming audio services (like Spotify) present a roadblock in that we cannot record directly from the other service. We would need to have the audio provided as files in a common format (MP3, AAC, FLAC).<br>This would also create copyright issues when shared online; Youtube is renowned for having DMCA requests filed because a user has used copyrighted material as the backing track of their video – sometimes it has simply been playing in the background.<br>It would be possible for users registered with Last.fm to hear preview clips of their most-listened songs but given that Last.fm has a fairly low user base and that the selectio of previews might seem disjointed from the visual content (or worse, distasteful combinations could occur), we would suggest that this isn’t worth it.<br>Final output videos<br>We will render a master copy in HD format and provide a lower-resolution file for sharing. |

## ESTIMATED BUDGET AND TIMESCALES

This is a rough estimate for budget and timescales for delivering the project, based on average UK agency rates. We would estimate that the project will take 3 – 4 months elapsed time.

| **Developer** | **Days** | **Rate** | | |
|---|---|---|---|---|
| | | | | |
| Server and Database Config and Architecture | 7 | 450 | | **£3,150** |
| | | | | |
| Web User Interface | 5 | 450 | | **£2,250** |
| | | | | |
| Data Ingest | 5 | 450 | | **£2,250** |
| | | | | |
| Automatic Data Ranking | 15 | 450 | | **£6,750** |
| | | | | |
| Manual / Preview Ranking | 10 | 450 | | **£4,500** |
| | | | | |
| Content Selection | 5 | 450 | | **£2,250** |
| | | | | |
| Output Process and Mastering | 7.5 | 450 | | **£3,375** |
| | | | | |
| Creating Effects with Animator | | | | |
| | | | | |
| | | | | |
| **Production, & Design** | | | | |
| | | | | |
| UI Design | 7.5 | 350 | | **£2,625** |
| | | | | |
| Producer / Project Manager | 15 | 450 | | **£6,750** |
| | | | | |
| | | | | |
| **Fixed Costs** | | | | |
| | | | | |
| Hosting, Hardware and Bandwidth Costs | | | | **£2,400** |
| | | | | |
| Third-party software | | | | |
| | | | | |
| | | Total | | **£36,300** |
| | | | | |
| | | Contingency | | **£3,630** |
| | | | | |
| | | VAT | | **£7,986** |
| | | | | |
| | | Grand Total | | **£47,916** |

# APPENDIX A: Potential Third-Party Video Solutions

Animoto
http://www.animoto.com

This solution was discounted because it does not offer third-party access.

Stupeflix
http://studio.stupeflix.com/en/ (also see: http://stupeflix-api.readthedocs.org/en/latest/tutorials/02_stupeflix_xml_langage.html and this about their integration with Brightcove: http://www.brightcove.com/en/partners/stupeflix)

This solution is the only viable solution for third-party software that we identified in the research. We would not recommend the third-party integration with Brightcove during the prototype as Brightcove is over-specified for the needs of this project.

Replay App
http://replayapp.com/

This solution only runs on mobile apps and does not offer third-party access to its systems.

Flixtime
http://www.flixtime.com

This solution was discounted because we were never able to log in (over the course of a few days), which would imply it is unreliable.

# APPENDIX B: An Alternative Approach HTML5 Video Experience

This document was designed to assess the feasibility of the requirements as stated to us. However, in assessing the requirements, we felt that there might be a stronger way to present the output of the project to the user. We felt that it might be more effective – and present the user with better quality video – to create an HTML5 video experience.

Two examples of these types of experience are the *Facebook Look Back* (https://www.facebook.com/help/www/206982576163229) and *Take This Lollipop* (http://www.takethislollipop.com/). In these cases, the content is presented through the native web browser, rather than in a composited video. This means that the final output can be reviewed from different points of view and in a more dynamic fashion.

The upside of this approach is that there is less video compression and compositing, which means the experience can run faster and at higher quality. The downside is that the number and sophistication of effects and transitions may be lower.

We have outlined how this could work below and can expand on it if desired.

## Core mechanism

Graph databases differ from traditional databases in that instead of a static schema to force data to specific patterns, they are interested in items and relationships.

An item is simply a piece of data with attributes. Users, posts, social networks, are all items with their own set of attributes depending on what they are. A graph database makes sense of these items by creating relationships between items. An item can have any number of relationships to any number of items. By defining items and relationships, we can build up an exploratory world for users.

## Building the experience

We would begin by selecting a protagonist and allowing the user to view items that applied to specified filters:

- Date
- Type (video, photo, tweet)
- Relationship (tagged in, posted by)
- Location

By providing these filters and relationships we can allow the user to move from item to item, around the timeline and in the social space.

## Technologies employed

All of this can be built natively in newer browsers using HTML5 technologies. This includes javascript implementations of graph databases*, HTML5 audio and video players.

It can be reformatted easily depending on the screen size, and therefore is well suited to all connected devices (not just PC/laptop).

Being interactive and two-way (i.e. we can find out who the viewer is) we can use information to change the experience based on location, relationship to the user and so on.

* To see a visual example of a graph database, see the demonstrator built for the GRAIN Photo Archive at the Library of Birmingham, visit: http://entertheswarm.com/?p=885