UCL DEPARTMENT OF
COMPUTER SCIENCE

Research Note
RN/14/12

Life and Death in the App Store: Theory and Analysis of Feature Migration

Federica Sarro, Mark Harman, Yue Jia, William Martin, Yuanyuan Zhang

University College London

{f.sarro, mark.harman, yue.jia, w.martin, yuanyuan.zhang}@ucl.ac.uk

Abstract

In this paper we introduce a feature migration theory and use it to study the migratory behaviour of 1,324 Blackberry features, finding that 68% die, 2% migrate and 30% are intransitive (they neither die nor migrate). Intransitive features have significantly higher prices ($p<0.05$), and higher popularity, while customers appear to be less sensitive to their price. By contrast, migratory features have lower price, rating and popularity and higher price sensitivity. We also introduce the Category Similarity Graph, based on a similarity metric which may help developers to better prepare for migration, because there is strong linear correlation ($rho= 0.62$, $p<0.001$) between two categories' similarity metric and their subsequent propensity for migration.

1 Introduction

App stores are vibrant software development marketplaces that offer software engineering developers and researchers a rich and varied source of information (prices, ratings and popularity). We study of app store feature migration, leveraging this information to provide insights into the way different features behave; some spread, some remain, some relocate and some die out. We introduce a set-theoretic formal characterisation of these migratory behaviours of features through app stores and use it to empirically investigate feature migration in the non-free feature space of the Blackberry app store.

We believe that the study of features' migratory trends can help us to understand the overall app store ecosystem [1] and relationships between its categories of apps. Such feature migration analysis may also offer benefits to developers, helping them to identify interesting and potentially important features from among the many thousands claimed by their peers and competitors. App developers may be interested in questions about feature migration and their relationship to the categories into which they release their apps.

For example, here are three illustrations of the kinds of question developers may ask (and our migration analysis answers) together with the reasons why developers might care about the answers:

Which migratory behaviours carry monetary value? Developers might pay attention to including higher priced features to add to their potential income.

Which migratory behaviours involve more popular features? Developers might care more about such popular features, since their customers appear to like them more.

Which categories are more likely to migrate features to one another? Developers might find technical opportunities for the symbiotic shared development of sets of apps in these categories.

For the purposes of this study we define a feature as follows: a feature is a claimed functionality offered by an app, captured by a set of collocated words in the app description and shared by a set of apps. In this paper we study the migration of these claimed features in the Blackberry app store. However, our formal characterisation of feature migration and its analysis are applicable to any app store and to any kind of feature (and feature extraction technique), thereby supporting widespread comparison of results across app stores and feature types.

We measure the price, rating and popularity (rank of downloads) of these features in terms of their averages (both mean and median) calculated over all apps that share the features [2]. This allows us to investigate empirically the differences and relationships between these attributes of features.

We also investigate relationships between app store categories in terms of feature migration between them. We found, perhaps unsurprisingly, that app categories that share a large number of features also have greater migration between them. We introduce definitions of category distance measures in terms of shared features and migration between categories, allowing us to map features'

migratory paths between categories and to better understand the relationship between categories. The primary contributions of this paper are as follows:
1. Theory: We introduce and formalise concepts of migration, exodus, extinction and intransitivity using a set theoretic formalism that casts all features into a subsumption hierarchy of migratory behaviours and the relationships between them.
2. Behavioural Differences: We present the results of an analysis of the Blackberry app store between Week 3 and Week 36 of the Year 2011. Our findings reveal that different migratory behaviours exhibit significantly different price, rating and popularity and also markedly different (strong and significant) correlations between them.
3. Depicting and Predicting Migrations: We introduce the Category Similarity Graph (CSG) and the Feature Migration Graph (FMG) to depict migratory behaviours and identify categories that may be symbiotic, because they exchange features. Our results show a strong linear correlation between the edge values of the two graphs, indicating that category similarity may be used to predict future feature migration.

Section 2 provides background information on our approach to mining app store repositories. Section 3 introduces our Set-Theoretic Theory of Feature Migration. Section 4 explains our empirical study design while Section 5 presents the results of the study and Section 6 discusses the threats to validity. Section 7 considers the related work, and Section 8 concludes.

2 Background

This section briefly reviews our approach to extracting feature information from app stores. More details are provided elsewhere [2, 3]. Our approach to app store analysis consists of four phases shown in Figure 1. The first phase extracts raw data from the app store (in this case BlackBerry App World[1], though our approach can be applied to other app stores with suitable changes to the extraction front end). The second phase parses the raw data extracted in the first phase to retrieve all the available attributes of each app relating to price, rating and textual descriptions of the app itself. The third phase analyses app descriptions to identify the features claimed for apps by their developers.

Phase 1 uses a customised web crawler to collect raw data from the app store, from which we parse the HTML to extract the descriptions and other data (rating, price and popularity, measured in terms of the rank of downloads) in Phase 2. This extraction process cannot be entirely automated, because some attribute fields (populated by humans) may need (human) disambiguation. For example, the price field contains entries like '0', 'Free', 'Free for one week' or a word that means 'free' in a language other than English, all of which signify that the app is provided without charge to the customer (at least initially). However, apart from this manual disambiguation step the process is fully automated.

[1] `http://appworld.blackberry.com/webstore/`

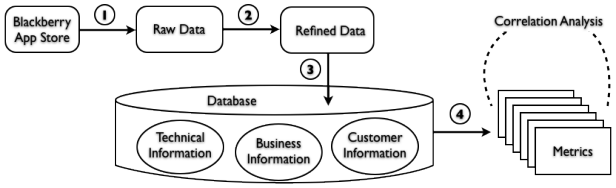


Figure 1: **Overall App Analysis Architecture**: A four phase approach extracts, refines and stores app information for subsequent analysis.

Phase 3 uses natural language processing to extract, from each description, the features claimed for the app by its developers. Such feature claims can be written in many ways by developers. We developed a four-step NLP algorithm to extract feature information and implemented it using the Natural Language Toolkit (NLTK) [4]. The first step extracts raw feature patterns, thereby identifying the 'coarse features' of apps. We locate raw feature patterns by searching for an HTML list in the description of apps. If the sentence prior to an HTML list contains at least one keyword from the set of words "include, new, latest, key, free, improved, download, option, feature", the HTML list is saved as the raw feature pattern for this app.

Non-English and numerical characters are removed along with unimportant English language STOPWORDS such as {'the', 'and', 'to'}. The words that remain are transformed into 'lemma form' using the WORDNETLEMMATIZER function from NLTK, thereby homogenising singular/plural, gerund endings and other non-germane grammatical details.

From this lemmatised, stop-word-reduced token stream, the algorithm extracts a set of 'featurelets'; a set of commonly occurring co-located words, identified using NLTK's N-gramCollocationFinder package. We use a greedy hierarchical clustering algorithm to aggregate all similar featurelets together. The algorithm initially treats each featurelet as a single cluster and, then, repeatedly combines clusters that are more than 50% similar. The result is a set of feature descriptions consisting of either 2 or 3 keywords (which we call 'bitri-grams') that describe the claimed feature.

We use a set of metrics that compute the rating, price and popularity of a feature in terms of the median value of the corresponding ratings, prices and popularities of all apps that possess the feature. We used the median, because app popularity is measured as an ordinal rank (called 'rank of downloads' by Blackberry) and the rating is a star rating (recorded for each app as a value from zero to 5 stars in half star increments). These two measurements are clearly ordinal scale measurements [5] and so the median is the most suitable average computation. For price, the use of median (instead of mean) for value aggregation is more questionable.

We did observe ordinal pricing behaviour. For example, the app store requires developers to charge in whole dollar increments. Furthermore, prices chosen by developers tend to cluster around ten, twenty and thirty dollar 'price

points', suggesting some kind of implicit 'ordinal scale' properties. However, the scale could equally well be argued to be a ratio scale.

In order to check that our choice of median aggregation did not affect the results we report here, we re-computed all results using mean to aggregate over app prices, ratings and popularities (these results are reported in Appendix). The findings remained as reported here, suggesting that the choice of aggregation technique is relatively unimportant for the features studied. For completeness, we provide all of our data on the accompanying website[2].

3 A Set Theoretic Characterisation of App Store Feature Migration

We are interested in features that migrate, because movement of features between categories suggests that these features have some form of transferable value beyond the category of apps in which they initially emerge in the app store ecosystem [1]. In order to define migration, we need to describe, first, the categories in which a feature resides at a given time in a given app store database. We define this formally as follows:

Definition 1 (Category Membership)**.** If a feature f in an app store database D is a member of category C at time t then we shall write $f \in C_{D\{t\}}$. We define the set of categories, $\mathcal{C}^f_{D\{t\}}$, of which a feature f is a member at time t in D, by extension, as $\{C \mid f \in C_{D\{t\}}\}$.

There are various behaviours that could be termed 'migratory'. We start with the weakest possible notion of migration, according to which a feature migrates if it resides in at least one new category at the end of the time period considered. More formally, we define the weak migration predicate on features as follows:

Definition 2 (Weak Migration)**.** A feature f in an app store database D (weakly) migrates between time t_0 and t_1, written $\mathcal{WM}^f_{D\{t_0,t_1\}}$ if and only if $\mathcal{C}^f_{D\{t_1\}} - \mathcal{C}^f_{D\{t_0\}} \neq \emptyset$.

We use set comprehension notation, $\{t_0, t_1\}$, for the time period from t_0 to t_1 to allow our theory to be more conveniently extended to multiple time periods, though we restrict ourselves to a single period in the analysis in this paper. Our definition of migration is termed '*weak* migration' because *any* newly entered category counts as a migration, even if the feature disappears from (some or all of) the categories from which it is migrating. If a feature does not weakly migrate, written $\mathcal{NM}^f_{D\{t_0,t_1\}}$, then it does not enter any new categories over the time period considered.

We also define *strong* migration, where a feature strictly spreads from at least one category to at least one new category (and remains in all categories in which it originated). More formally, we define strong migration as follows:

[2] `http://www0.cs.ucl.ac.uk/staff/F.Sarro/projects/UCLappA/home.html`

Definition 3 (Strong Migration)**.** A feature f in an app store database D strongly migrates between time t_0 and t_1, written $\mathcal{SM}^f_{D\{t_0,t_1\}}$ if and only if (iff)

$$\begin{array}{l}(\mathcal{C}^f_{D\{t_0\}} - \mathcal{C}^f_{D\{t_1\}} = \emptyset) \ \wedge \\ (\mathcal{C}^f_{D\{t_0\}} \cap \mathcal{C}^f_{D\{t_1\}} \neq \emptyset) \ \wedge \\ (\mathcal{C}^f_{D\{t_1\}} - \mathcal{C}^f_{D\{t_0\}} \neq \emptyset)\end{array}$$

That is, a strongly migratory feature has no categories that it abandons $(\mathcal{C}^f_{D\{t_0\}} - \mathcal{C}^f_{D\{t_1\}} = \emptyset)$ and at least one category in which it remains $(\mathcal{C}^f_{D\{t_0\}} \cap \mathcal{C}^f_{D\{t_1\}} \neq \emptyset)$ and at least one new category that it spreads into $(\mathcal{C}^f_{D\{t_1\}} - \mathcal{C}^f_{D\{t_0\}} \neq \emptyset)$.

A feature that strongly migrates also weakly migrates, but not necessarily *vice versa*, hence the choice of terminology (strong and weak).

A specific category of weak migration, which we term 'exodus', is also worthy of definition. There are also weak and strong forms of exodus. In a weak exodus, a feature disappears from *at least one* of the categories in which it previously resided, while appearing (for the first time) in at least one new category. In a strong exodus, a feature disappears from *all* categories in which it previously resided to take up residence in at least one new category. More formally:

Definition 4 (Weak Exodus)**.** A feature f in an app store database D experiences weak exodus between time t_0 and t_1, written $\mathcal{WE}^f_{D\{t_0,t_1\}}$, iff $\mathcal{WM}^f_{D\{t_0,t_1\}} \wedge \neg(\mathcal{SM}^f_{D\{t_0,t_1\}})$.

Definition 5 (Strong Exodus)**.** A feature f in an app store database D experiences strong exodus between time t_0 and t_1, written $\mathcal{SE}^f_{D\{t_0,t_1\}}$, iff $\mathcal{WE}^f_{D\{t_0,t_1\}} \wedge (\mathcal{C}^f_{D\{t_0\}} \cap \mathcal{C}^f_{D\{t_1\}} = \emptyset)$.

Our definitions are so-construed that weak migration captures all possible migratory behaviours. It is the union of those features that strongly migrate and those that weakly exodus (which, in turn, includes those that strongly exodus).

There is a special case of strong exodus, permitted by our definitions, in which a feature appears for the first time at the end of the time period considered. That is, such a feature resides in *no* categories at the start of the time period (so $\mathcal{C}^f_{D\{t_0\}} = \emptyset$) and is in at least one new category at the end of the time period (so $\mathcal{C}^f_{D\{t_1\}} \neq \emptyset$). This situation is a special case of strong exodus, a feature's 'birth', in which it undergoes an 'exodus into the app store from nowhere'.

In our empirical analysis that follows, we do not include the 'Birth' of features, since we wish to focus on migration of existing features through the app store. However, for completeness, we define the Birth category, formally, as follows:

Definition 6 (Birth)**.** The Birth of feature f in an app store database D between time t_0 and t_1, written $\mathcal{B}^f_{D\{t_0,t_1\}}$, occurs iff $\mathcal{SE}^f_{D\{t_0,t_1\}} \wedge \mathcal{C}^f_{D\{t_0\}} = \emptyset$.

All of the migratory behaviours we describe and formalise involve some form of change in the categories in which the feature resides, except one, which we term the 'intransitive' features. An intransitive feature neither appears in any new categories nor does it disappear from any between the start and the end of the time period considered. More formally, we define intransitivity as follows:

Definition 7 (Intransitive)**.** A feature f in an app store database D is intransitive between time t_0 and t_1, written $\mathcal{I}^f_{D\{t_0,t_1\}}$ iff

$$\begin{array}{l}(\mathcal{C}^f_{D\{t_0\}} - \mathcal{C}^f_{D\{t_1\}} = \emptyset) \ \wedge \\ (\mathcal{C}^f_{D\{t_0\}} \cap \mathcal{C}^f_{D\{t_1\}} \neq \emptyset) \ \wedge \\ (\mathcal{C}^f_{D\{t_1\}} - \mathcal{C}^f_{D\{t_0\}} = \emptyset)\end{array}$$

That is, an intransitive feature has no categories that it abandons ($\mathcal{C}^f_{D\{t_0\}} - \mathcal{C}^f_{D\{t_1\}} = \emptyset$) and at least one category in which it remains ($\mathcal{C}^f_{D\{t_0\}} \cap \mathcal{C}^f_{D\{t_1\}} \neq \emptyset$) and it has no categories to which it spreads ($\mathcal{C}^f_{D\{t_1\}} - \mathcal{C}^f_{D\{t_0\}} = \emptyset$).

If a feature neither migrates, nor remains intransitive then it must be dying out (either from some or all categories) which we term 'extinction' in this paper. Once again, there is a strong and a weak form of extinction. In a weak extinction, the feature disappears from *at least one* category in which it resided and does not migrate to any new ones. In a strong extinction, a feature completely disappears; it disappears from *all* categories in which it resided and does not migrate to any new ones. More formally, we define weak and strong extinction as follows:

Definition 8 (Weak Extinction)**.** A feature f in an app store database D experiences weak extinction between time t_0 and t_1, written $\mathcal{WX}^f_{D\{t_0,t_1\}}$, iff $\mathcal{NM}^f_{D\{t_0,t_1\}} \wedge \neg(\mathcal{I}^f_{D\{t_0,t_1\}})$.

Definition 9 (Strong Extinction)**.** A feature f in an app store database D experiences strong extinction between time t_0 and t_1, written $\mathcal{SX}^f_{D\{t_0,t_1\}}$, iff $\mathcal{WX}^f_{D\{t_0,t_1\}} \wedge \mathcal{C}^f_{D\{t_1\}} = \emptyset$.

There is special case of strong extinction, in which no category contains the feature of interest, so $\mathcal{C}^f_{D\{t_0\}} = \mathcal{C}^f_{D\{t_1\}} = \emptyset$.

In this situation the feature is not in the app store at the start, nor at the end, of the time period considered: it is *unborn*, or equivalently we might say that 'it is *undead*'. That is, though the feature may exist *outside* the app store time period considered, it does not exist in the app store within the period considered. Without meaning to become unreasonably philosophical (or worse, supernatural), we might say that a feature that does exist in a previous time period is 'undead', while one that does not is 'unborn'. We make this distinction in the interests of theoretical completeness; it has no further bearing on the study on which we report.

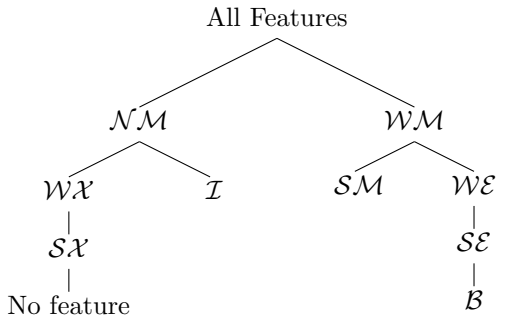


Figure 2: The Theoretical Feature Migration Subsumption Hierarchy

As can be seen, our definitions are loosely analogous to animal migration terminology, where features are analogous to animals and categories to geographic regions. These definitions of the different kinds of migratory behaviour form the set-theoretic subsumption relationship depicted in Figure 2. The theory is also complete; it captures all possible features in a single subsumption hierarchy of behaviours with respect to the birth, migration and extinction of features.

To see that this theory captures all possible features and to help visualise each, consider the Venn diagram in Figure 3 and the associated mapping of all possible set configurations and their corresponding migratory definitions in Table 1. This subsumption relationship allows us to speak formally and precisely about features movement through the app store in terms of their birth, migration and death. It also precisely captures the relationships between the different kinds of feature movement that we observe in practice. We call this feature movement 'migratory behaviour'.

Having explored the relationships between and within different forms of migratory behaviour, we turn to the implications for migration on the categories of apps between which features may migrate. There is a natural measure of feature similarity between app categories: the similarity between two categories is the normalised size of their shared feature set. More formally, we define category similarity as follows:

Definition 10 (Jaccard Similarity)**.** We define the similarity between two categories, C_1 and C_2 in an app store database D at time t as the normalised size of the shared feature set between the categories:

$$\frac{\#(C_{1D\{t\}} \cap C_{2D\{t\}})}{\#(C_{1D\{t\}} \cup C_{2D\{t\}})}$$

We define a Category Similarity Graph (CSG):

Definition 11 (Category Similarity Graph (CSG))**.** The CSG for an app store database D at a time t is an undirected graph, in which the nodes are the categories of D at t, and there is an edge between every pair of categories, C_1 and C_2, labelled with their Jaccard Similarity, $\mathcal{JD}_{D_t}(C_1, C_2)$. When visualising

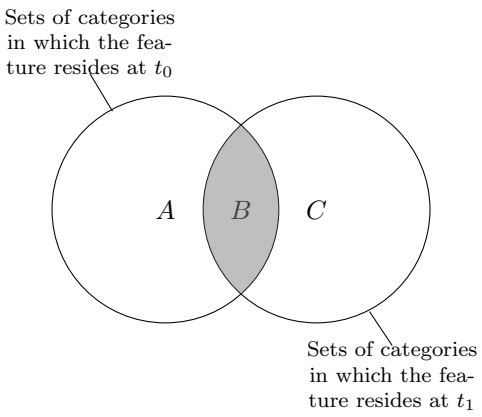


Figure 3: **Venn diagram showing the sets of categories a feature resides in at both snapshots.** A = Categories that have the feature at t_0 but not t_1. C = Categories that have the feature at t_1 but not t_0. B = Categories that have the feature at both t_0 and t_1. Each of A,B and C could be empty or not, so we have 8 possibilities (shown in Table 1, with their corresponding definitions).

Set			**Meaning**
Migratory behaviours (Weakly Migrating ($\mathcal{WM}$)):			
A	B	C	**Behaviour**
0	1	1	Strongly Migration ($\mathcal{SM}$)
–	$\bar{0}1$	–	Weak Exodus ($\mathcal{WE}$)
-	0	1	Strong Exodus ($\mathcal{SE}$)
0	0	1	Birth ($\mathcal{B}$)
Non-Migratory behaviours (Not weakly Migrating ($\mathcal{NM}$)):			
A	B	C	**Behaviour**
0	1	0	Intransitive ($\mathcal{I}$)
–	$\bar{0}1$	–	Weak Extinction ($\mathcal{WX}$)
-	0	0	Strong Extinction ($\mathcal{SX}$)
0	0	0	No feature (unborn or undead)

Table 1: **Completeness of Migratory Definitions.** Set names (A, B, and C) refer to the sets in the Venn diagram (Figure 3). Necessarily empty sets are denoted by 0. Necessarily non-empty sets by 1. The entry '-' indicates sets which are unconstrained. The entry '– $\bar{0}1$ –' indicates that sets A and B are unconstrained *except* it cannot be that *both* A is empty *and* B is non-empty. As can be seen, the two rows labelled in this way therefore capture all possibilities not covered by the row immediately above them in the table.

CSGs, we may picture only a subset of edges, such as those with above (chosen) threshold similarity.

We also define a Feature Migration Graph, which captures the migratory paths (and the number of features that travel along them) between the categories of an app store:

Definition 12 (Feature Migration Graph (FMG))**.** The FMG for an app store database D at a time t is a directed graph, in which the nodes are the categories of D at t, and there is an edge from category C_1 to each other category and C_2, labelled with the size of the set of features that migrate from C_1 to C_2, according to one of our definitions of migration.

The FMG concerns migration from a specific category to another so, in computing edge labels, we restrict attention to the two categories that participate in the migration. For example, in the FMG for Weak Migration, each edge is labelled with $\#(\{f \mid \mathcal{WM}^{f}_{(D\dagger\{C_1,C_2\})\{t_0,t_1\}}\})$, where $X \dagger Y$ is the app store database X restricted to the categories in the set Y.

4 Empirical Study Design

This section explains our empirical study design and motivates our research questions and the statistical tests we use.

4.1 Dataset

We extracted data from the Blackberry app store at two time points (Week 3 and Week 36 in 2011). Table 2 presents summary data for these two 'snapshots'. The choice of time points for this first investigation of feature migration is partly arbitrary, since any two time points could be used to illustrate migration. However, we wanted to select two time points that were sufficiently separated that we might reasonably expect some changes, yet not so far apart that any migratory behaviour observed could not reasonably be acted upon by developers. Thus, we selected two points within the same year, but separated by 33 weeks. Future work will explore other time granularities to identify the smallest and largest time periods over which migration can be meaningfully observed.

4.2 Research Questions

Clearly there is little value to be gained from investigating feature migration if there is no change between the time points considered; all features would simply be intransitive according to our definitions. This motivates our first research question, which establishes whether there is change and, if there is, how much change is found within each category. Since this research question is simply a 'sanity check' and not a particularly important finding in its own right, we number it 'zero':

Table 2: Summary Data for the Blackberry apps Studied Between Two Time Intervals (Week 3 and Week 36 in 2011).

	2011-week03										2011-week36									
Category	Apps	Features	Mean Price	Median Price	Mean Rating	Median Rating	Mean Rank of Downloads	Median Rank of Downloads	Min Rank of Downloads	Max Rank of Downloads	Apps	Features	Mean Price	Median Price	Mean Rating	Median Rating	Mean Rank of Downloads	Median Rank of Downloads	Min Rank of Downloads	Max Rank of Downloads
Business	205	82	14.81	4.99	1.86	2.00	8337	8108	803	18201	348	77	12.61	4.99	1.79	0.00	19250	18183	807	42585
Education	163	47	10.88	4.99	1.29	0.00	9526	9674	715	18320	592	80	5.66	2.99	1.38	0.00	22535	22682	1608	42673
Entertainment	456	106	4.99	2.99	1.82	2.00	7363	6622	97	18253	920	85	6.28	2.99	1.87	1.00	18593	16697	135	42628
Finance	107	77	5.42	3.99	2.10	2.00	7552	6196	168	17963	194	73	4.50	2.49	1.93	1.25	19842	16863	257	41787
Games	1633	49	3.54	2.99	1.91	2.00	7343	6432	165	18312	2618	35	2.64	1.99	2.14	2.50	16087	13730	156	42635
Health & Wellness	379	100	17.26	3.99	1.32	0.00	9045	9106	220	18077	632	87	15.76	3.99	1.57	0.00	20058	18562	258	42260
IM & Social Networking	78	67	5.13	2.99	1.78	2.00	6670	5046	19	18217	152	69	4.12	1.99	2.43	3.00	14992	11843	23	41924
Maps & Navigation	140	67	11.20	3.99	1.89	2.00	7167	5812	639	18126	284	69	12.40	9.99	1.98	2.00	18382	16066	664	42653
Music & Audio	94	69	4.51	2.99	1.75	1.50	8132	6653	142	18236	512	81	2.02	0.99	1.01	0.00	24882	27532	208	42620
News	43	38	3.36	2.99	1.33	0.50	9271	9018	1165	16151	75	42	2.31	0.99	1.75	1.00	17864	15640	1402	41957
Photo & Video	80	101	4.34	2.99	2.36	2.50	5180	3324	8	18273	423	91	2.48	1.99	1.34	0.00	22118	24710	16	42644
Productivity	334	87	8.49	4.99	2.37	3.00	6688	5692	124	18315	506	82	6.21	2.99	2.59	3.00	15023	11824	259	42643
Reference & eBooks	4356	89	5.73	2.99	0.14	0.00	13869	14491	1151	18319	11597	77	4.26	0.99	0.12	0.00	30759	31570	1181	42663
Shopping	22	61	4.31	2.99	2.20	2.50	6064	4066	676	15878	45	53	2.68	1.99	2.11	2.50	15896	11866	2585	37814
Sports & Recreation	172	35	6.31	2.99	1.73	1.00	8991	8614	216	18272	254	37	4.90	2.99	1.93	1.00	19577	16791	954	42651
Themes	4481	34	3.63	2.99	1.87	0.00	9326	9601	88	18314	11131	28	3.11	2.99	1.69	0.00	21347	21543	19	42674
Travel	450	70	6.95	5.99	0.55	0.00	11827	11986	1124	18309	769	79	4.77	2.99	0.66	0.00	25798	26477	558	42671
Utilities	715	69	5.04	2.99	2.15	2.50	6938	6021	32	18239	1377	66	4.64	2.99	2.31	2.50	16549	14267	63	42642
Weather	41	76	8.43	9.99	2.28	2.50	5364	5074	198	13629	60	66	7.39	5.99	2.40	2.50	13051	10786	311	42045
Mean	734	70	7.07	3.99	1.72	1.47	8140	7449	408	17758	1710	67	5.72	3.12	1.74	1.17	19611	18296	603	42219
Median	172	69	5.42	2.99	1.86	2.00	7552	6622	198	18239	506	73	4.64	2.99	1.87	1.00	19250	16791	259	42635

RQ0. Feature Evolution: Is there any change in features between the start and end time points?

We answer RQ0 simply by measuring the number of features in each category at the start and end of the time period. We also compute the Jaccard Similarity between the initial and final versions of each category. Assuming we do observe a change in each category's number of features over the time period, then the next natural question to ask is whether each of the migratory behaviours we defined theoretically, also exists in practice. If it does, what is the distribution of features over the subsumption hierarchy of migratory behaviours. This motivates RQ1:

RQ1. Feature Migration: How do the features distribute over the different migratory behaviours in the subsumption hierarchy?

If we find that our theoretical migratory behaviours exist in practice, then this is intellectually interesting, but it is only of practical significance if we also observe important differences in the price, rating or popularity of different kinds of migratory behaviour. This motivates RQ2:

RQ2. Are there any significant differences in the price, rating, popularity of features that exhibit different migratory behaviours?

We use a 2-tailed, unpaired Wilcoxon test [6] to compare the median values of the price, rating and popularity of each of the migratory behaviours. We use the Wilcoxon test because we are investigating ordinal data and therefore need a non-parametric statistical test, with fewer assumptions about the underlying data distribution. The test is 2-tailed because there is no assumption about which median will be higher, and it is unpaired, because there are different numbers of features exhibiting each behaviour. In our case, the Wilcoxon test is identical to the closely-related Mann Whitney 'U' test [7], which could also be used with identical results.

The Null Hypothesis is that there is no difference in price (respectively rating or popularity) between categories. In common with most scientific inferential

statistical testing, we set the significance level 95%, so that we have only a 0.05 probability of committing a Type 1 error (incorrectly rejecting the Null Hypothesis). This choice is justified by the fact that rejection of the Null Hypothesis would be a finding that would lead to actionable conclusions. That is, developers should start to measure and take note of migratory behaviours in app stores. Therefore, we require relatively strong evidence to support such findings. Since we perform multiple statistical tests we also use the Bejamini-Hochberg correction [8] to ensure that we retain only a 0.05 probability of Type 1 error.

If there is a significant difference between the price, rating or popularity of features that exhibit different migratory behaviours then we also investigate the statistical effect size of the difference using the Vargha-Delaney $\hat{A}_{12}$ metric for effect size (as recommended by Arcuri and Briand [9]). Like the Wilcoxon test, the Vargha-Delaney $\hat{A}_{12}$ makes few assumptions and is suited to ordinal data such as ours. It is also highly intuitive: for a given feature attribute (price, rating or popularity), $\hat{A}_{12}(A, B)$ is simply an estimate of the probability that the attribute value of a randomly chosen feature from migratory behaviour group A will be higher than that of migratory behaviour group B.

Over the whole Blackberry app store we previously observed [2] that there is a correlation between rating and popularity: higher rated features are more popular than lower rated features (they have lower ranks of download, indicating that they are more frequently downloaded). However, there was no such correlation for price (and either rating or popularity). This raises the natural question as to whether the correlations observed over the whole app store are mirrored within the features that share each form of migratory behaviour. Alternatively, a perhaps more intriguing find would be that certain forms of migration also come with their own specific properties, as expressed through observations of correlations between the three attributes of price, rating and popularity. This motivates RQ3:

RQ3. Are there differences in the correlations between price, rating and popularity within each form of migratory behaviour?

In order to study this question we use both the Pearson [10] and Spearman statistical correlation tests [11]. While the Pearson rho value assesses the degree of linear correlation, the Spearman rho value assesses the degree of rank correlation. A rho value of 1 indicates perfect correlation, while -1 indicates perfect inverse correlation. A value of zero indicates no correlation. Absolute values between 0 and 1 indicate the degree of correlation (or inverse correlation) present. Different interpretations can be placed on the rho values reported for linear and rank correlation. However, we may conservatively state that there is some evidence of a correlation when the absolute rho value is greater than 0.5 and strong evidence when rho is greater than 0.7. Both are also reported with a p value that denotes the probability that reported rho value is different to zero (no correlation).

Strictly speaking, since our data is measured on an ordinal scale, findings reported using the Pearson correlation should be treated with a degree of caution. However, as previously observed (in Section 2) there are grounds for consider-

ing price to be a ratio scale measurement, so Pearson correlations may be more intuitively applied in this case (as well as Spearman rank correlations).

Having explored feature migration between categories, we now turn to the relationship between categories of app and the features that migrate to and from these categories. We might speculate that two categories that share a large normalised overlap in features enjoyed by their apps would also experience a greater degree of migration.

RQ4. Is migration more prevalent between similar categories?

We answer RQ4 by measuring the similarity of each category, and constructing the corresponding CSG for the 19 categories of the Blackberry app store. We finally measure the number of features that migrate to and from each category to construct the Feature Migration Graph (FMG).

We then rank category pairs (the edges of these two graphs) by their edge labels (similarity and total features migrating) and investigate the correlation between them.

5 Results

RQ0. Feature Evolution: Table 3 reports the number of features contained in the Blackberry app store at two different period of times (i.e., weeks 3 and 36 of the year 2011, denoted T_0 and T_1 respectively) and the Jaccard Similarity (JS) of each category over the time (i.e., we measure how the features contained in the same category change over the time). The total number of features decreases slightly over the two snapshots (from 1,360 to 1,316).

More importantly, as can be seen the JS value is far from 1.0 in all cases, so there is a great deal of change to be studied: some of the features must die or migrate, motivating the rest of our analysis.

RQ1. Feature Migration: According to the definitions given in Section 3, we augment the Subsumption Hierarchy with the number of features found in each category (see Figure 5). As the figure shows, we found that 1,292 features do not migrate and 32 features do. This is an encouraging finding for app store developers: it means that if we also find that migratory features have interesting properties, then they are also sufficiently few in number that they could be tracked and considered in some detail.

Of the 1,292 non migratory features, we found that 394 were Intransitive (I), remaining unmoved, while 884 completely die out becoming Strongly Extinct (SX), and a further 14 partly die out (becoming weakly extinct but not strongly extinct). Of the 32 migratory features, we found that 12 Strongly Migrated (SM) to different categories, while 20 left at least one of the original category to exodus to new ones (WE), with 15 of these abandoning all their previous categories to migrate to new ones (SE).

Table 4 reports the 12 Strongly Migratory (SM) features. We can observe that these features always migrate to a category that has similar characteristics. As can be seen from their bitri-gram names, most of these features have clearly

Table 3: RQ0. Number of features contained in a given category and Jaccard Similarity (JS) of the initial and final categories over the time period.

Category	T_0	T_1	JS
Business	82	77	0.17
Education	47	80	0.26
Entertainment	106	85	0.12
Finance	77	73	0.18
Games	49	35	0.29
Health & Wellness	100	87	0.07
IM & Social Networking	67	69	0.18
Maps & Navigation	67	69	0.34
Music & Audio	69	81	0.19
News	38	42	0.27
Photo & Video	101	91	0.05
Productivity	87	82	0.31
Reference & eBooks	89	77	0.20
Shopping	61	53	0.39
Sports & Recreation	35	37	0.24
Themes	34	28	0.22
Travel	70	79	0.13
Utilities	69	66	0.18
Weather	76	66	0.25
Total	1,324	1,277	-

'transferable value' that could cross category boundaries (e.g., easy-access, add-list, latest-news). We report all 12, but developers may choose to focus on only a subset of interest. Three of the strongly migratory features originate in the 'Maps and Navigation' category and offer location aware functionality, underscoring the importance of context aware features in mobile apps.

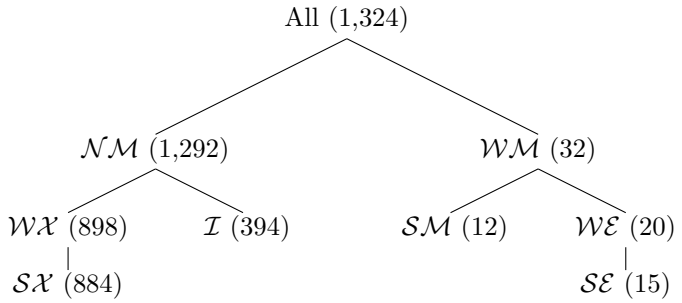


Figure 4: RQ1. Observed Number of Features for Each Migratory Behaviour.

Table 4: RQ1. The Strongly Migratory Features.

Feature	Initial Category	Spreads to Category
$[add, list]$	Shopping	Productivity
$[application, icon]$	Entertainment	Themes
$[current, location]$	Maps & Navigation	Travel
$[detailed, map]$	Maps & Navigation	Travel
$[easy, access]$	Reference & eBooks	Education
$[icon, set]$	Themes	Entertainment
$[latest, news]$	News	Sports & Recreation
$[location, find]$	Maps & Navigation	Travel
$[one, time]$	Business	Utilities
$[score, game]$	Games	Sports & Recreation
$[screen, device]$	Reference & eBooks	Entertainment

RQ2. Differences in Migratory Behaviours: Figure 5 show the boxplots of the Median Price, Rating and Rank of Downloads values of the features that have the same migratory behaviours[3]. This figure reveals a surprising finding: though migratory features clearly have functionality to offer that transcends the categories into which the feature was originally deployed, they also have a lower rating, popularity and price than the non-migratory features. It seems that developers should take account of these features (since they can apply in multiple categories, perhaps allowing for code re-use and cross-category development), but they cannot expect to be rewarded by higher income, popularity and ratings for including them.

Perhaps a more encouraging finding for app developers lies in the 394 intransitive features, which remain within a category and neither die out nor migrate. Manual inspect of these features confirmed that they seem to refer to category specific functionality. Examples are 'forecast-current-condition' (in the Weather category) and 'automatically-save-game' in the Games category. We find evidence that these intransitive features do carry higher monetary value. Also, since they show no sign of dying out, they are perhaps more worthy of the

[3]The boxplots of the Mean Price, Rating and Rank of Downloads values are reported in Appendix B .

investment in developers' time to recoup this income.

We investigated the differences in the price, rating and popularity using inferential statistical tests, but since there are relatively few features that are migratory, these findings were not conclusive. Therefore, we have only weak evidence that price, rating and popularity are lower for migratory features, though perhaps sufficient to motivate future work on this question (naturally, we make all data and analysis available for further study on the paper's companion website[4]).

For the relatively larger category of intransitive features, the Wilcoxon test revealed a significant difference between the price of $\mathcal{I}$ and its counterpart in the non-migratory category ($\mathcal{WX}$) ($p = 0.001, \hat{A}_{12} = 0.56$ and $p = 0.007, \hat{A}_{12} = 0.55$, for mean-based and median-based feature price computation respectively). There is also a significant difference between the price of $\mathcal{I}$ and $\mathcal{SX}$ ($p < 0.001, \hat{A}_{12} = 0.56$ and $p = 0.007, \hat{A}_{12} = 0.55$, for mean-based and median-based feature price computation respectively). The detailed results of the Wilcoxon test can be found in Appendix B .

In conclusion, to answer RQ2, we find that the intransitive features are significantly higher priced than the other non-migratory features and that there is some tentative evidence that suggests that migratory features, though inherently important, may also be lower rated, less popular and cheaper than non-migratory features.

RQ3. Correlations among Price, Popularity and Rating: Table 5 presents the Pearson and Spearman correlations for the raw data (based on scatter plots of each pair of {Price, Popularity, Rating} values for each feature[5]). We only report the correlation coefficient (rho value) where the p value indicates that the correlation coefficient is reliable (i.e., we have evidence that it is significantly different to zero). Where the $p > 0.05$ we leave the entry blank, since there are insufficiently many data points to allow us to draw reliable conclusions about correlations.

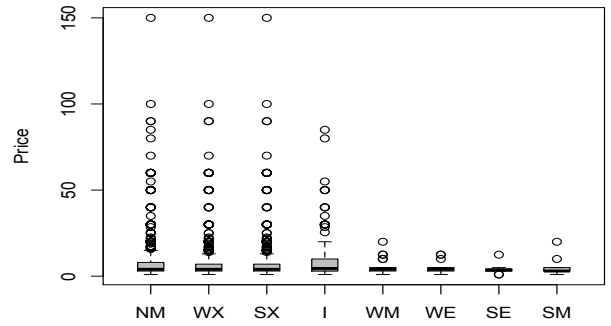
As previously observed for the app store as a whole [2], we find a strong correlation between rating and popularity for all eight forms of migratory and non-migratory behaviour[6]. However, as Table 5 reveals, there is evidence of a strong inverse correlation between price and each of rating and popularity (reverse rank of downloads) was observed for the strongly migratory features ($\mathcal{SM}$).

This correlation is not present in the raw data for features as a whole. It indicates that the more expensive a strongly migratory feature, the lower its rating and popularity. Other correlation coefficients are significant (so there is evidence that they have at least a 0.95 probability of being non-zero), but are not nearly as strong.

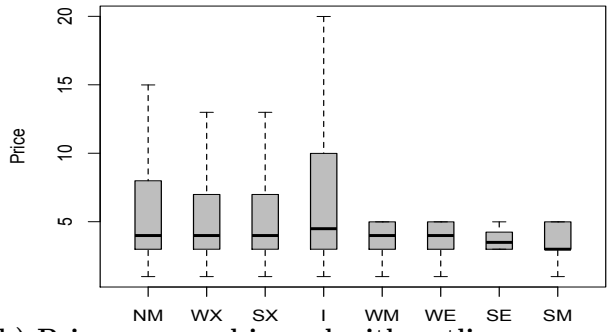
[4] `http://www0.cs.ucl.ac.uk/staff/F.Sarro/projects/UCLappA/home.html`

[5] These scatter plots can be found in Appendix C .

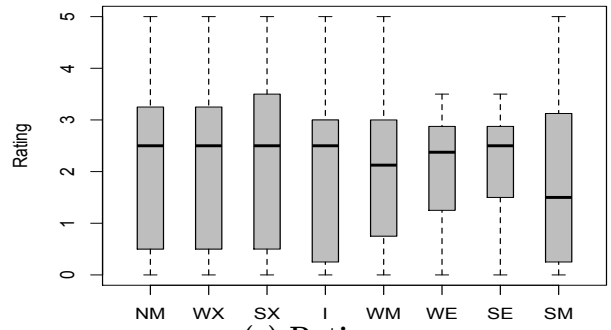
[6] Since this analysis concerns a different snapshot of the app store state to previous work [2], this finding is therefore a replication of the previous finding that rating and popularity are strongly correlated in the Blackberry app store.



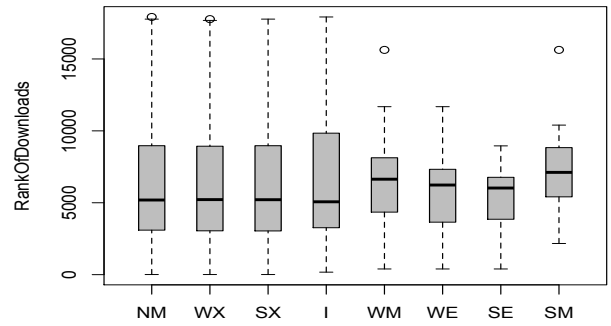
(a) Price



(b) Price - zoomed in and with outliers removed



(c) Rating



(d) Popularity (Rank of Downloads)

Figure 5: RQ2. Boxplots of Price, Rating and Popularity (Rank of Downloads) for each of the non-migratory behaviours. The first four boxplots of each figure are non-migratory, while the second four are migratory. A higher Rank of Downloads indicates lower popularity. It is interesting to note that migratory features are lower rated and less popular, yet they colonise new categories. Most striking of all, the strongly migratory features which carry most transferable value, spreading through the app store, are also the cheapest, least popular and lowest ranked features. Also, importantly for app developers, the intransitive features carry the highest monetary value; notably higher than either those features that migrate or those that die out.

Table 5: RQ3. Raw Value Correlations. Pearson and Spearman Correlation values for (P)rice, (R)ating and Rank of (D)ownloads. Only significant correlation values ($p \leq 0.05$) are reported.

	Pearson						Spearman					
Migratory Behaviour	Mean PR	Median PR	Mean PD	Median PD	Mean RD	Median RD	Mean PR	Median PR	Mean PD	Median PD	Mean RD	Median RD
$\mathcal{NM}$	-0.30	0.30	0.34	0.34	-0.80	-0.81	-0.19	-0.20	0.21	0.20	-0.79	-0.77
$\mathcal{WX}$	-0.31	-0.31	0.36	0.35	-0.78	-0.79	-0.19	-0.20	0.22	0.20	-0.77	-0.75
$\mathcal{SX}$	-0.31	-0.31	0.35	0.35	-0.78	0.79	-0.18	-0.18	0.21	0.21	-0.77	-0.77
$\mathcal{I}$	-0.26	-0.27	0.30	0.32	-0.84	-0.85	-0.18	-0.17	0.19	0.20	-0.83	-0.80
$\mathcal{WM}$					-0.80	-0.74					-0.83	-0.79
$\mathcal{SM}$	-0.74		0.76	0.77	-0.82	-0.65	-0.79	-0.61	0.66	0.51	-0.85	-0.80
$\mathcal{WE}$					-0.84	-0.86					-0.84	-0.84
$\mathcal{SE}$					-0.64	-0.69					-0.76	-0.72

Since prices are charged at price points (in whole dollar increments), we can also compute the median rating (respectively rank of downloads) for all features that share a given price point. When we do this over all features, we observe a correlation between the price point and both the median rating (R) and the median rank of downloads (D) [3]. We also investigate whether this correlation is observed for each of the migratory behaviours. Table 6 reports the results.

Because we are summarising a set of data points (those that share a price point) as a single median value, we reduce the number of data points, and therefore reduce the evidence on which to draw conclusions about correlations. However, where there is significant evidence for a correlation, the trend is clear.

The significant correlation observations provide further evidence that there is price sensitivity for migratory features (the observation that higher prices correlate to lower popularity is even stronger for them). It also provides further evidence for the potential attractiveness to developers of the intransitive features: there appears to be notably less price sensitivity to these features. That is, the inverse correlation between price and both rating and popularity is notably weaker for the intransitive features compared to all features and to the other features, which either tend to die out or migrate.

Table 6: RQ3. Median Price Point Correlations. Pearson and Spearman correlation values for median (R)ating and Rank of (D)ownloads for each price point. For completeness, all migratory behaviours are listed in the rows of the table. However, only significant correlation values ($p \leq 0.05$) are reported.

	Pearson Median		Spearman Median	
Migratory Behaviour	PR	PD	PR	PD
$\mathcal{WE}$				
$\mathcal{SM}$		-0.88		
$\mathcal{SE}$		-0.75		
$\mathcal{WX}$	-0.51	-0.62	-0.60	-0.64
$\mathcal{I}$	-0.49	-0.52	-0.51	-0.40
$\mathcal{SX}$	-0.51	-0.62	-0.60	-0.64
All features	-0.57	-0.65	-0.67	-0.62

RQ4. Category Neighbours: Figure 6 depicts the Feature Migration Graph (FMG) for the migratory features found in our study. As can be seen, all 19 categories participated at least one migration over the time period we considered. However, not all the categories export features to other categories: two categories (i.e., Education and Sports & Recreation) only receive incoming migrations. We speculate that features developed for these categories tend to be more specific, offering developers less 'transferable value' for their development effort.

Most categories are involved in one-to-one migration: a feature moves to only one new category. However, 9 of the categories export features to more than one other category. These categories are Business, Entertainment, Health & Wellness, IM & Social Networking, Maps & Navigation, Music & Audio, Reference & Books, and Weather. The developers of apps that target these categories are perhaps particularly fortunate to have potential to multiply the re-use of the features they develop in other categories.

Perhaps most interesting to developers would be the 4 bidirectional edges which denote mutual sharing of features between categories (i.e., Business-Utilities, Business-Health & Wellness, Maps & Navigation-Weather and Entertainment-Themes). These edges suggest pairs of categories in which the most symbiotic software development can take place. We have evidence from the migration of features between them to indicate that effort spent on development in each of the symbiotic pair can benefit development in its partner category.

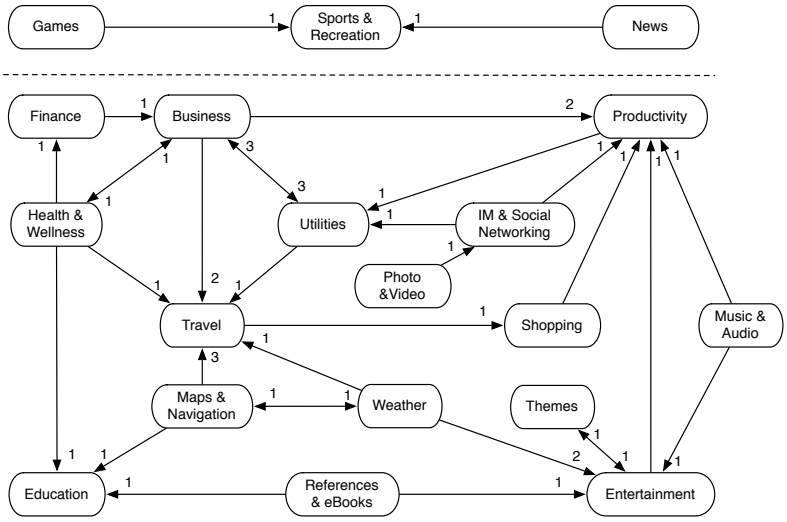


Figure 6: RQ4.The Feature Migration Graph (FMG).

Table 7 reports the departure and arrival categories for migration, together with the number of features that migrate, the number of features that these categories shared before the migration (T_1), and their category similarity (JS).

In order to investigate whether developers could use our category similarity measure to identify likely symbiotic categories in which shared development could be mutually beneficial, we calculated the correlation between the FMG and the CSG. We found a strong positive correlation between the similarity of two categories and the subsequent number of features that migrate between them (Pearson $rho = 0.62, p < 0.001$), indicating that developers can predict which categories are more likely to migrate to each other.

Table 7: RQ4. Departure and Arrival Categories for the Migratory Features and their Category Similarity.

From Category	To Category	Migrated	Shared	JS
Business	Travel	2	0	0.000
Business	Utilities	3	7	0.042
Business	Productivity	2	3	0.017
Business	Health & Wellness	1	2	0.011
Entertainment	Productivity	1	1	0.005
Entertainment	Themes	1	2	0.014
Finance	Business	1	1	0.006
Games	Sports & Recreation	1	0	0.000
Health & Wellness	Education	1	0	0.000
Health & Wellness	Finance	1	0	0.000
Health & Wellness	Business	1	2	0.011
Health & Wellness	Travel	1	0	0.000
IM & Social Networking	Productivity	1	0	0.000
IM & Social Networking	Utilities	1	2	0.013
Maps & Navigation	Travel	3	1	0.007
Maps & Navigation	Entertainment	1	0	0.000
Maps & Navigation	Productivity	1	0	0.000
Music & Audio	Entertainment	1	0	0.000
Music & Audio	Productivity	1	0	0.000
News	Sports & Recreation	1	2	0.026
Photo & Video	IM & Social Networking	1	0	0.000
Productivity	Utilities	1	2	0.012
Reference & Books	Education	1	0	0.000
Reference & Books	Entertainment	1	0	0.000
Shopping	Productivity	1	2	0.013
Themes	Entertainment	1	2	0.014
Travel	Shopping	1	0	0.000
Utilities	Business	3	7	0.042
Utilities	Travel	1	1	0.007
Wheather	Entertainment	2	1	0.005
Wheather	Travel	1	0	0.000
Wheather	Maps & Navigation	1	0	0.000

6 Threats to Valitidy

Threats to External Validity: Though our feature migration theory is general, our empirical results are specific to the two snapshots of the BlackBerry App World we consider and more work would be required to investigate whether the findings generalise to other time periods and app stores.

Internal Validity Threat Risk Reduction: The inferential statistical values and correlations reported in this paper and all derived metrics reported were independently computed by two different authors, and cross-checked.

Threats to Construct Validity: We measure only features reported by app store developers in the apps' descriptions, and make no claim to measure features in the code of the apps. Strictly speaking, this is not a threat to construct validity, since we believe that developers' technical claims about their apps are an interesting kind of feature in their own right.

7 Related and Future Work

There is much more work that can be done to further understand the concept of feature migration in app stores (as software ecosystems [1]). Migratory features are interesting because of the many possibilities that they suggest for future work. They may be interesting for refactoring: perhaps such features would make useful library components. They are also the 'ones to watch' because they potentially apply to more apps than the developer may have realised. This section briefly summarises this work and its relationship to our findings and the possible avenues for future work it opens up.

The goal of App Store Analysis [12, 13, 14, 15, 16] is to combine technical data with non-technical data such as user and business data to understand their inter-relationships. App stores provide feedback in the form of user reviews. Many authors have focused their analysis on this aspect of the app store [17, 18, 19, 20, 21, 22, 23]. Iacob and Harrison [19] report that 23.3% of the reviews they studied were found to be feature request, further underscoring the importance of features in app store ecosystems.

One natural extension of our work would be to investigate the interplay between feature migration and user requests. Pagano and Maalej [22] also found that review feedback was correlated with higher ratings and that most reviews appears very soon after a new version of an app is released. This offers the hope that developers could react to feature requests, perhaps particularly targeting likely migratory features in a timely fashion.

We extract features from the descriptions of apps uploaded to the app store by developers. Therefore, when we speak of a 'feature', we are speaking about a claimed feature; a feature that the developers claim to offer in their app description. Other authors have studied other features, in various forms, that exist in the code itself and also the relationship between feature claims in descriptions and features found in apps. For example, Gorla et al. [24] use API calls to detect aberrant or otherwise suspicious behaviour. Pandita et al. [25] compare

the permissions requested by the app and the app description, thereby identifying suspect descriptions. Yang et al., [26] also considered this problem, using topic modelling (whereas Pandita et al. used first order logic). Another natural step for future work would be to examine the way these kind of features migrate through app stores, and whether there is a relationship between migration of claims and migration of code.

Despite this recent explosion in activity in App Store Analysis, no previous work has considered the movement of features in app stores. In order for us to capture this feature movement (which we call migration), we need to consider the status and app store at different snapshots, taken at different times during the evolution of the app store. To the best of the author's knowledge, no previous work has considered any form of analysis over more than one 'snapshot' of the app store state. However, we believe the future work may find many other possible applications and implications for such 'longitudinal' studies of app stores over periods of time.

8 Conclusion

We have introduced a theory and study of feature migration in app stores. Overall, we find that a relatively small proportion of features are migratory (only approximately 2% of all features). An even smaller proportion is strongly migratory in the sense that they spread throughout the app store categories without vacating any categories in which they previously resided. Indeed, there are sufficiently few such strongly migratory features, that the developers could reasonably find time to study them in some detail.

We present evidence to suggest that developers have reason to be interested in both migratory and intransitive features. Though strongly migratory features are inherently important (since they cut across many category boundaries) they carry less value to developers (since they are cheaper) and also have lower than average ratings and popularity. There is also evidence that customers are more price sensitive to migratory features. Many features (approximately 68%) tend to die out and the developer will naturally be less interested in these; why waste time on features that are 'here today and gone tomorrow?'. By contrast, the features that neither migrate nor die out, which we term the 'intransitive features' (about 30% of all features) appear to be of great value to developers: they have higher than average price and also attract higher ratings and popularity. Furthermore, there is evidence to suggest that customers are less price sensitive to intransitive features.

We also found evidence that the Category Similarity Graph that we introduce may help developers to understand (and perhaps to prepare for) likely migration between categories, because there is a strong linear correlation between category similarity and subsequent propensity for future migration between categories. Developers can use this information to identify symbiotic categories in which development effort can be reduced by mutual feature sharing.

We believe that our results, taken together, provide compelling evidence that

feature migration is both interesting to researchers and potentially valuable to developers. We also believe that other longitudinal studies, involving multiple snapshots of app store state may reveal similar interesting behaviours, both for apps and features that they offer.

Acknowledgment

The research is funded by Engineering and Physical Sciences Research Council CREST Platform Grant (EP/G060525) and Dynamic Adaptive Automated Software Engineering (DAASE) programme grant (EP/J017515).

References

[1] S. Jansen, M. A. Cusumano, and S. Brinkkemper, *Software Ecosystems: Analyzing and Managing Business Networks in the Software Industry*. Edward Elgar publishing Ltd., 2013.

[2] M. Harman, Y. Jia, and Y. Zhang, "App store mining and analysis: MSR for App Stores," in 9^{th} *Working Conference on Mining Software Repositories (MSR 2012)*, Zurich, Switzerland, 2-3 June 2012.

[3] A. Finkelstein, M. Harman, Y. Jia, W. Martin, F. Sarro, and Y. Zhang, "App store analysis: Mining app stores for relationships between customer, business and technical characteristics," Department of Computer Science, University College London, Research Note RN/14/10, 2014.

[4] E. Loper and S. Bird, "NLTK: The Natural Language Toolkit," in *Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics (TeachNLP '02)*. Philadelphia, USA: Association for Computational Linguistics, 7-12 July 2002, pp. 69–72.

[5] M. J. Shepperd, *Foundations of software measurement*. Prentice Hall, 1995.

[6] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[7] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Annals of Mathematical Statistics*, vol. 18, no. 1, pp. 50–60, 1947.

[8] Y. Bejamini and Y. Hochberg, "Controlling the false discovery rate: A practical and powerful approach to multiple testing," *Journal of the Royal statistical Society (Series B)*, vol. 57, no. 1, pp. 289–300, 1995.

[9] A. Arcuri and L. Briand, "A practical guide for using statistical tests to assess randomized algorithms in software engineering," in 33^{rd} *International Conference on Software Engineering (ICSE'11)*. New York, NY, USA: ACM, 2011, pp. 1–10.

[10] K. Pearson, "Notes on regression and inheritance in the case of two parents," *Proceedings of the Royal Society of London*, vol. 58, pp. 240–242, June 1895.

[11] C. E. Spearman, "The proof and measurement of association between two things," *The American Journal of Psychology*, vol. 15, no. 1, pp. 72–101, January 1904.

[12] T. Menzies, "Beyond Data Mining; Towards "Idea Engineering"," in *Proceedings of the 2nd International NSF sponsored Workshop on Realizing Artificial Intelligence Synergies in Software Engineering (RAISE '13)*, San Francisco, USA, 25-26 May 2013.

[13] R. Minelli and M. Lanza, "Software Analytics for Mobile Applications - Insights & Lessons Learned," in *Proceedings of the 17th European Conference on Software Maintenance and Reengineering (CSMR '13)*. Genova, Italy: IEEE, 5-8 March 2013.

[14] M. Nagappan, E. Shihab, and A. E. Hassan, "Challenges in mobile apps: A multi-disciplinary perspective," in *Conference of the Center for Advanced Studies on Collaborative Research (CASCON '13)*, 2013, pp. 378–381.

[15] I. Ruiz, M. Nagappan, B. Adams, and A. Hassan, "Understanding reuse in the android market," in *Program Comprehension (ICPC), 2012 IEEE 20th International Conference on*, June 2012, pp. 113–122.

[16] M. D. Syer, M. Nagappan, A. E. Hassan, and B. Adams, "Revisiting prior empirical findings for mobile apps: An empirical case study on the 15 most popular open-source Android apps," in *Conference of the Center for Advanced Studies on Collaborative Research (CASCON '13)*, 2013, pp. 283–297. [Online]. Available: http://dl.acm.org/citation.cfm?id=2555523.2555553

[17] E. Guzman and W. Maalej, "How do users like this feature? a fine grained sentiment analysis of app reviews," in *Requirements Engineering (RE 2014)*, 2014, to appear; available on line.

[18] L. Hoon, R. Vasa, J.-G. Schneider, and J. Grundy, "An analysis of the mobile app review landscape: Trends and implications," 2014, available on line from Swinbourne University of Tethnology, Australia.

[19] C. Iacob and R. Harrison, "Retrieving and Analyzing Mobile App Feature Requests from Online Reviews," in *Proceedings of the 10th Working Conference on Mining Software Repositories (MSR '13)*, San Francisco, California, USA, 18-19 May 2013.

[20] H. Khalid, "On identifying user complaints of iOS apps," in *35th International Conference on Software Engineering (ICSE 2013)*, D. Notkin, B. H. C. Cheng, and K. Pohl, Eds. IEEE/ACM, 2013, pp. 1474–1476.

[21] H. Khalid, E. Shihab, M. Nagappan, and A. Hassan, "What do mobile app users complain about? A study on free iOS apps," *IEEE Software*, 2014, to appear; available online.

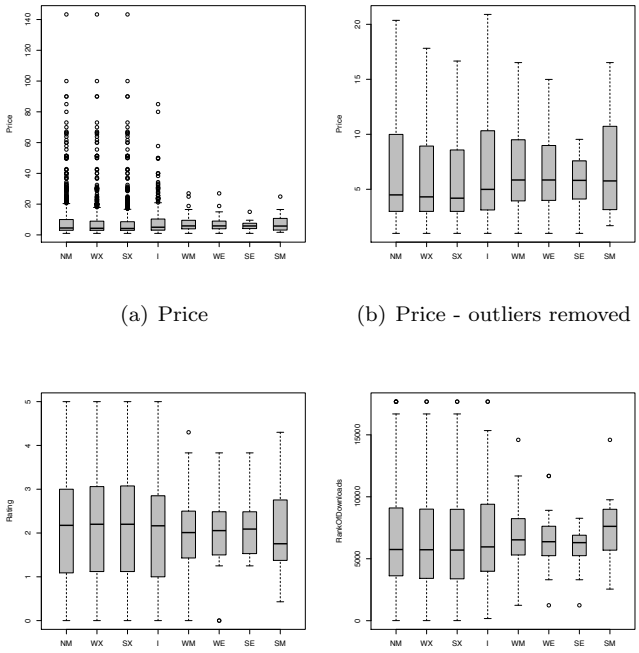
[22] D. Pagano and W. Maalej, "User feedback in the appstore: An empirical study," in *requirements engineering (RE 2013)*. IEEE, 2013, pp. 125–134. [Online]. Available: http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6621629

[23] I. J. M. Ruiz, M. Nagappan, A. Bra, T. Berger, S. Dienst, and A. E. Hassan, "On the relationship between the number of ad libraries in an android app and its rating," 2014, available on line from Queen's University, Canada.

[24] A. Gorla, I. Tavecchia, F. Gross, and A. Zeller, "Checking app behavior against app descriptions," in *36th International Conference on Software Engineering (ICSE 2014)*, 2014, pp. 1025–1035.

[25] R. Pandita, X. Xiao, W. Yang, W. Enck, and T. Xie, "WHYPER: Towards Automating Risk Assessment of Mobile Applications," in *Proceedings of the 22nd USENIX Security Symposium*, Washington DC, USA, 14-16 August 2013.

[26] Y. Yang, J. S. Sun, and M. W. Berry, "APPIC: Finding the hidden scene behinddescription files for android apps," 2014, available on line from University of Tennessee, USA. [Online]. Available: http://web.eecs.utk.edu/~jysun/research/publication/lda.pdf



(a) Price

(b) Price - outliers removed

(c) Rating

(d) Rank Of Downloads

Figure 7: RQ2. Boxplots of Mean Price, Rating and Popularity (Rank of Downloads) for each of the non-migratory behaviours.

A RQ2. Boxplots of Mean Price, Rating and Rank of Downloads

Figure 7 show the boxplots of the Mean Price, Rating and Rank of Downloads values of the features that have the same migratory behaviours. The first four boxplots of each figure are non-migratory, while the second four are migratory. A higher Rank of Downloads indicates lower popularity. The results confirm the ones obtained by using the median values (see Section 5).

B RQ2. Wilcoxon Test

Tables 8, 9, 10 and Tables 11, 12, 13 report the results of the Wilcoxon test obtained by comparing the Mean and Median Price, Rating and Rank of Dowloads of the considered migratory behaviours, respectively. Each table reports the p-value, the corrected p-value and the corresponding A^{12} effect size.

Table 8: Wilcoxon Test Results: mean price.

	SX			**WX**			**I**			**SM**			**WE**		
	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}
SX		-													
WX	0.898	1	0.502		-										
I	**0.001**	**0.004**	**0.559**	**0.001**	**0.006**	**0.557**		-							
SM	0.441	1	0.565	0.458	1	0.562	0.997	1	0.5		-				
WE	0.174	0.87	0.589	0.185	0.925	0.586	0.73	1	0.523	0.785	1	0.531		-	
SE	0.529	1	0.547	0.55	1	0.545	0.738	1	0.525	0.845	1	0.525	0.664	1	0.545

Table 9: Wilcoxon Test Results: mean rating of downloads.

	SX			**WX**			**I**			**SM**			**WE**		
	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}
SX		-													
WX	0.999	1	0.5		-										
I	0.164	0.821	0.524	0.163	0.814	0.524		-							
SM	0.716	1	0.531	0.711	1	0.531	0.947	1	0.506		-				
WE	0.454	1	0.549	0.455	1	0.549	0.697	1	0.526	0.953	1	0.508		-	
SE	0.83	1	0.516	0.83	1	0.516	0.888	1	0.511	0.678	1	0.55	0.764	1	0.532

Table 10: Wilcoxon Test Results: mean rank of downloads comparison among the migratory behaviours.

	SX			**WX**			**I**			**SM**			**WE**		
	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}
SX		-													
WX	0.903	1	0.502		-										
I	0.139	0.695	0.526	0.167	0.835	0.524		-							
SM	0.183	0.916	0.612	0.189	0.947	0.61	0.255	1	0.596		-				
WE	0.482	1	0.546	0.505	1	0.544	0.704	1	0.525	0.284	1	0.617		-	
SE	0.955	1	0.504	0.987	1	0.501	0.855	1	0.514	0.113	0.564	0.683	0.617	1	0.552

Table 11: Wilcoxon Test Results: median price.

	SX			WX			I			SM			WE		
	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}
SX		-													
WX	0.993	1	0.5		-										
I	0.007	0.034	0.547	0.007	0.033	0.547		-							
SM	0.614	1	0.542	0.609	1	0.543	0.262	1	0.595		-				
WE	0.879	1	0.51	0.878	1	0.51	0.282	1	0.571	0.617	1	0.554		-	
SE	0.341	1	0.571	0.338	1	0.571	0.078	0.39	0.633	0.919	1	0.514	0.444	1	0.577

Table 12: Wilcoxon Test Results: median rating.

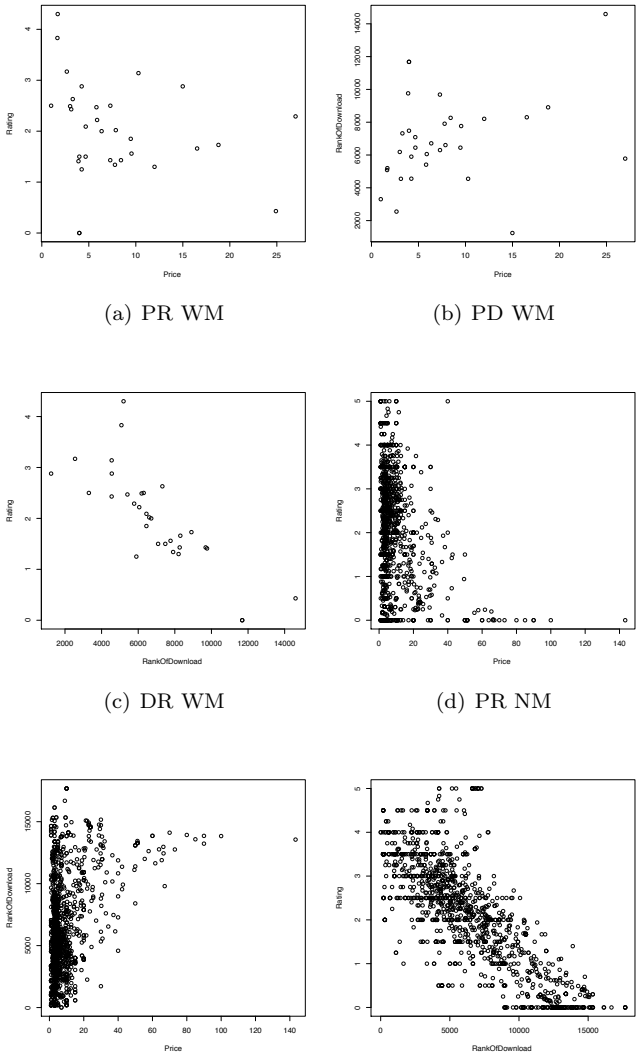
	SX			WX			I			SM			WE		
	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}
SX		-													
WX	0.974	1	0.5		-										
I	0.136	0.68	0.526	0.142	0.711	0.525		-							
SM	0.53	1	0.552	0.533	1	0.552	0.765	1	0.525		-				
WE	0.499	1	0.544	0.503	1	0.543	0.74	1	0.522	0.829	1	0.525		-	
SE	0.866	1	0.513	0.872	1	0.512	0.906	1	0.509	0.606	1	0.561	0.686	1	0.542

Table 13: Wilcoxon Test Results: median rank of downloads.

	SX			WX			I			SM			WE		
	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}	p	p_c	A^{12}
SX		-													
WX	0.935	1	0.501		-										
I	0.454	1	0.513	0.5	1	0.512		-							
SM	0.146	0.732	0.622	0.149	0.745	0.621	0.158	0.791	0.62		-				
WE	0.784	1	0.518	0.8	1	0.517	0.899	1	0.508	0.206	1	0.638		-	
SE	0.934	1	0.506	0.914	1	0.508	0.903	1	0.509	0.083	0.416	0.7	0.714	1	0.538

C RQ3. Scatter plots

Figures 8, 9, 10, 11, show the scatter plots of each pair of mean {Price, Popularity, Rating} values for each migratory behaviour.

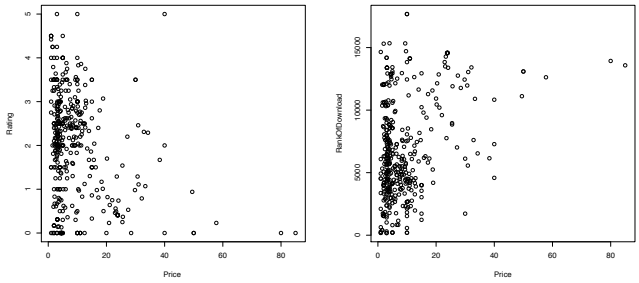


(a) PR WM (b) PD WM

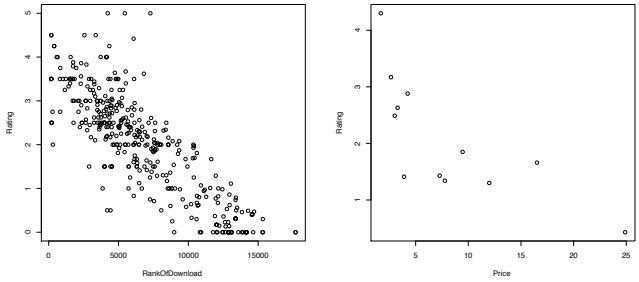
(c) DR WM (d) PR NM

(e) PD NM (f) DR NM

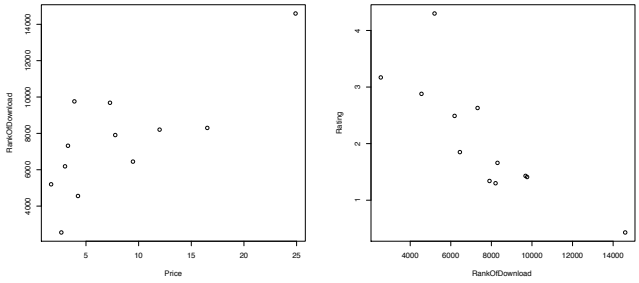
Figure 8: **RQ4**: Scatterplot of Mean Price (P), Rank of Downloads (D) and Rating (R) for the migratory behaviours (W)eak (M)igration and N(o)(M)igration



(a) PR I (b) PD I

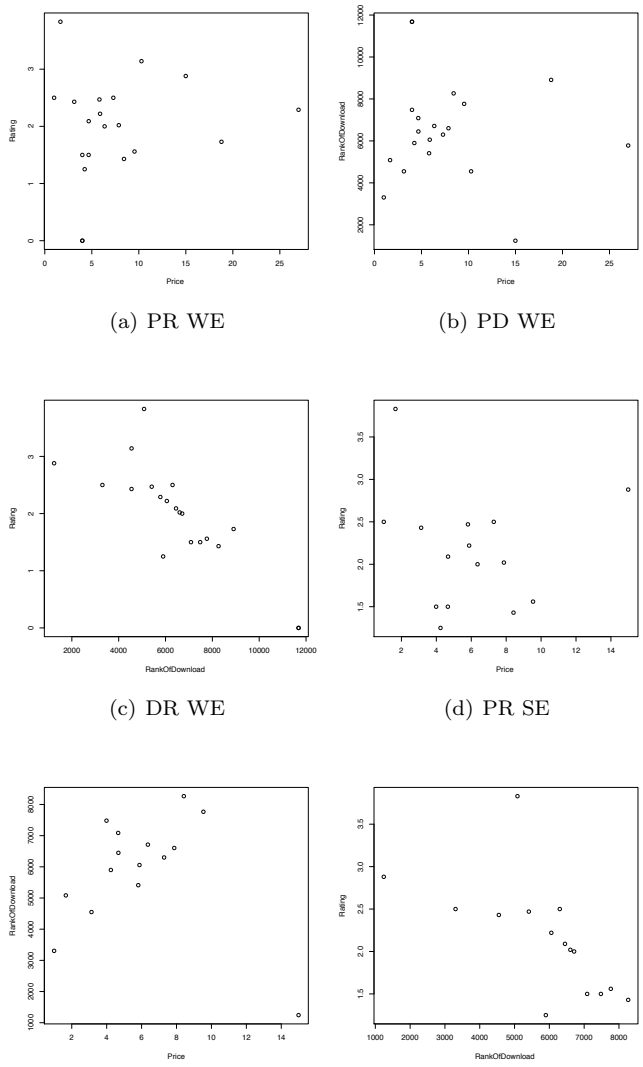


(c) DR I (d) PR SM



(e) PD SM (f) DR SM

Figure 9: **RQ4**: Scatterplot of Mean Price (P), Rank of Downloads (D) and Rating (R) for the migratory behaviours (I)ntransitive and (S)trong (M)igration.

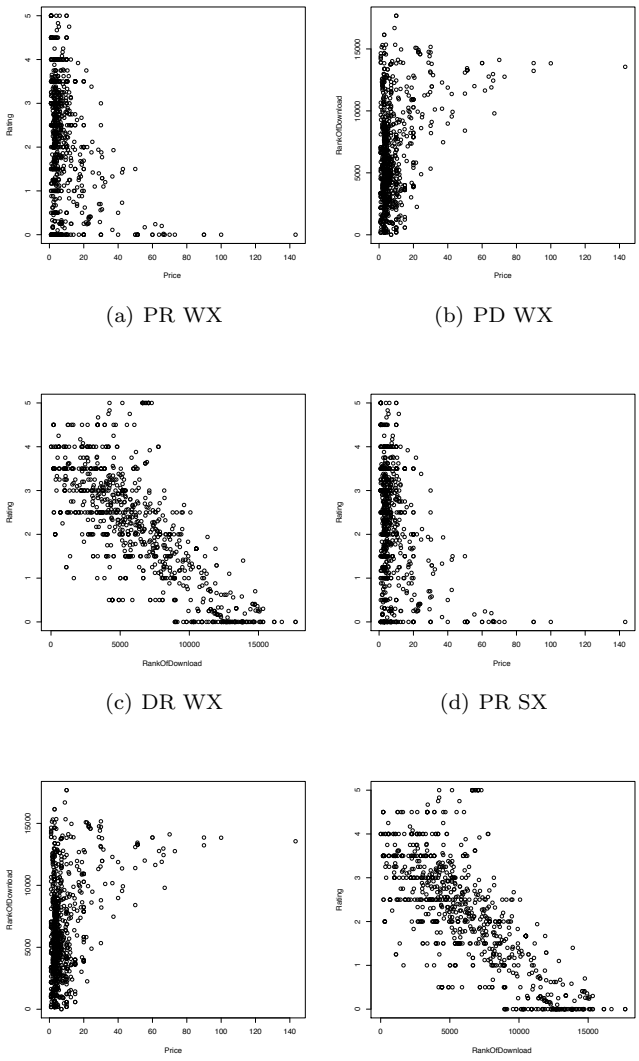


(a) PR WE (b) PD WE

(c) DR WE (d) PR SE

(e) PD SE (f) DR SE

Figure 10: **RQ4**: Scatterplot of Mean Price (P), Rank of Downloads (D) and Rating (R) for the migratory behaviours (W)eak (E)xodus and (S)trong (E)xodus.

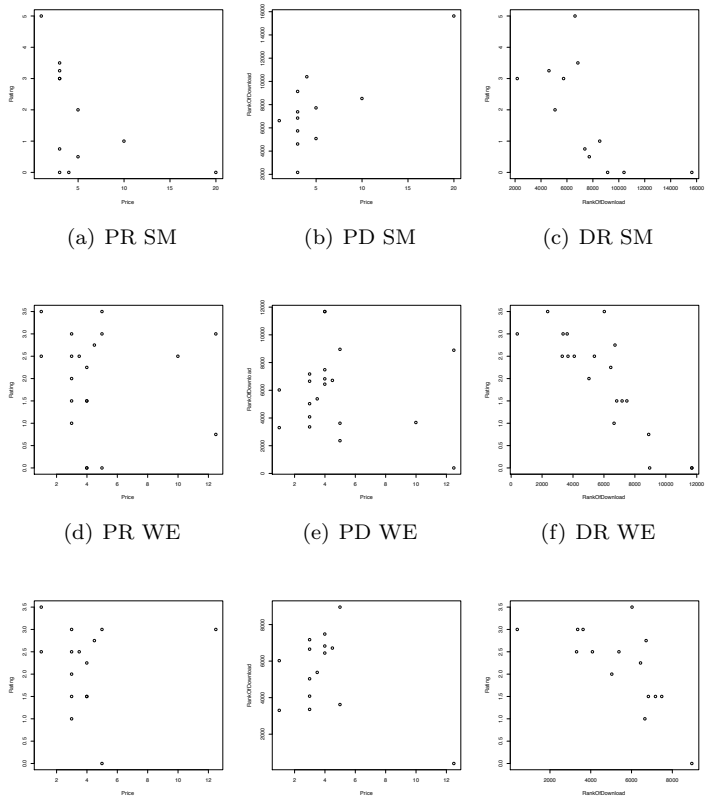


(a) PR WX (b) PD WX

(c) DR WX (d) PR SX

(e) PD SX (f) DR SX

Figure 11: **RQ4**: Scatterplot of Mean Price (P), Rank of Downloads (D) and Rating (R) for the migratory behaviours (W)eak e(X)tinction and (S)trong e(X)tinction.

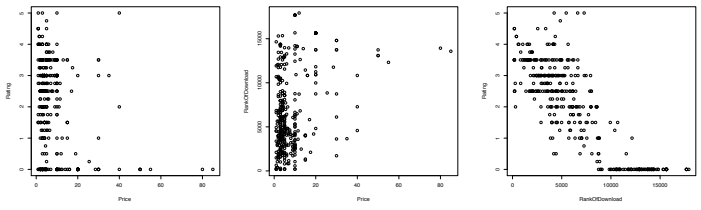


(a) PR SM (b) PD SM (c) DR SM

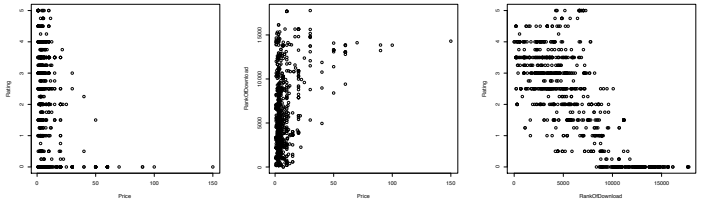
(d) PR WE (e) PD WE (f) DR WE

(g) PR SE (h) PD SE (i) DR SE

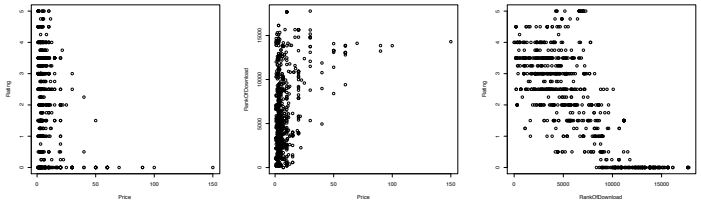
Figure 12: RQ4.Scatterplot of Median Price (P), Rank of Downloads (D) and Rating (R) for the migratory behaviours (S)trong (M)igration, (W)eak (E)xodus, (S)trong (E)xodus.



(a) PR I (b) PD I (c) DR I

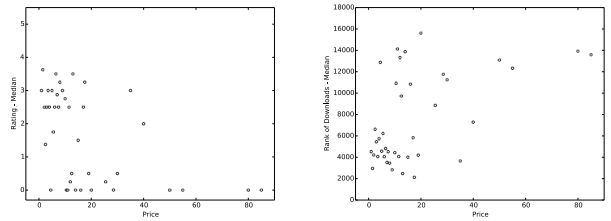


(d) PR WX (e) PD WX (f) DR WX



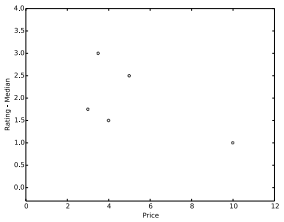
(g) PR SX (h) PD SX (i) DR SX

Figure 13: RQ4. Scatterplot of Median Price (P), Rank of Downloads (D) and Rating (R) for the migratory behaviours (I)ntranstitive, (W)eak e(X)tinction and (S)trong e(X)tinction

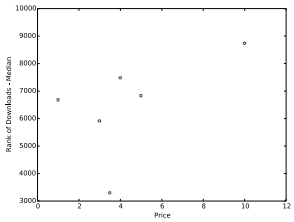


(a) PR I

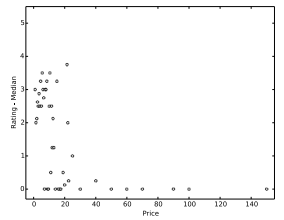
(b) PD I



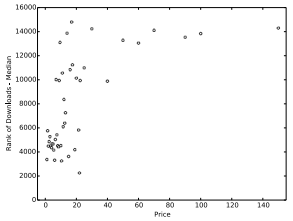
(c) PR WX



(d) PD WX



(e) PR SX



(f) PD SX

Figure 14: RQ4. Scatterplot of Median Price (P) and Rank of Downloads (D), and Median Price (P) Rating (R) for the migratory behaviours (I)ntransitive, (W)eak e(X)tinction and (S)trong e(X)tinction. Please, note that we grouped the points based on their median values.