Decision Support in Social Media and Cloud Computing

Vom Fachbereich Rechts- und Wirtschaftswissenschaften
der Technischen Universität Darmstadt

zur Erlangung des akademischen Grades
Doctor rerum politicarum (Dr. rer. pol.)

genehmigte

Dissertation

von Diplom-Wirtschaftsinformatiker
Jörg Gottschlich
aus Würzburg

Erstreferent:	Prof. Dr. Oliver Hinz
Zweitreferent:	Prof. Dr. Peter Buxmann
Tag der Einreichung:	01.03.2016
Tag der mündlichen Prüfung:	09.06.2016
Erscheinungsort/-jahr	Darmstadt 2016
Hochschulkennziffer	D17

Ehrenwörtliche Erklärung

Ich erkläre hiermit ehrenwörtlich, dass ich die vorliegende Arbeit selbstständig angefertigt habe. Sämtliche aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht.

Die Arbeit wurde bisher nicht zu Prüfungszwecken verwendet und noch nicht veröffentlicht.

Darmstadt, den 01. März 2016

Abstract

This cumulative dissertation examines applications of decision support in the field of social media and cloud computing. By the advent of Social Media, Big Data Analytics and Cloud Computing, new opportunities opening up in the field of decision support due to availability and ability to process new types of data sets. In this context, this dissertation introduces systems for the use of social media data for decisions and an approach for decision support in choosing a cloud computing provider.

In this dissertation, the benefits of different Facebook profile data for use in product recommender systems will be analyzed. Two experiments are carried out, in which the recommendation quality is determined by user survey. In another part of this dissertation, structured stock recommendations of an online community are used to automatically derive and update a stock portfolio. So investment decisions in the stock market are supported by a regular recalculation of the community rating for individual stocks. An succeeding article on this topic develops a formalized model for the description of investment strategies to enable a portfolio management system that automatically follows a strategy parameterized by an investor. Finally, a cloud broker model is presented which offers price / performance-based decision support in identifying an appropriate IaaS provider on the market for public cloud services. In a fundamental part of the thesis an IT architecture design is proposed which allows the parallel use and evaluation of different solution approaches in an operative IT system. Statistical tests are used to identify the best performing approach(es) and prefer them quickly while in operation. Overall, this cumulative dissertation consists of an introduction and five published articles.

Zusammenfassung

Diese kumulative Dissertation untersucht Anwendungsfälle von Entscheidungsunterstützung im Umfeld von Social Media und Cloud Computing. Durch das Aufkommen von Social Media, Big Data Analytics und Cloud Computing erschließen sich im Bereich der Entscheidungsunterstützung neue Möglichkeiten aufgrund Verfügbarkeit und Auswertbarkeit neuartiger Datenbestände. In diesem Rahmen stellt diese Dissertation neben Systemen zur Nutzung von Social-Media-Daten für Entscheidungen, auch Ansätze zur Entscheidungsunterstützung bei der Auswahl eines Cloud-Computing-Providers vor.

Zusammengefasst werden in dieser Arbeit anhand von Produktempfehlungen auf Basis von Facebookprofildaten der Nutzen der verschiedenen Profildaten für den Einsatz in Empfehlungssystemen analysiert. Dazu werden zwei Experimente durchgeführt, in denen die Empfehlungsqualität durch Nutzerbefragung ermittelt wird. In einem weiteren Teil der Arbeit werden strukturierte Aktienempfehlungen einer Online-Community zur automatisierten Gestaltung und Aktualisierung eines Aktienportfolios genutzt. So werden Investmententscheidungen am Aktienmarkt durch regelmäßige Neuberechnung der Community-Bewertung einzelner Aktien unterstützt. Ein weiterer Artikel entwickelt hierzu ein formalisiertes Modell zur Beschreibung von Anlagestrategien, so dass eine automatisierte Portfolioverwaltung durch ein System ermöglicht wird, die einer vom Investor parametrisierten Strategie folgt. Schließlich wird ein Cloud-Broker-Modell vorgestellt, das zu einem gegebenen Anwendungsfall eine preis-/leistungsbasierte Unterstützung bei der Identifizierung eines passenden IaaS-Providers am Markt für Public Cloud Services bietet. In einem grundlegenden Teil der Dissertation wird ein IT-Architekturdesign vorgeschlagen, das den parallelen Einsatz unterschiedlicher Lösungsansätze zur Evaluation in einem operativen IT-System ermöglicht und diese gegeneinander testet, um den besten Ansatz zu identifizieren und zu bevorzugen. Insgesamt besteht die kumulative Dissertation aus einer Einleitung und fünf bereits veröffentlichten Artikeln.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

ALM	Asset and Liability Management
API	Application Programming Interface
BA	Business Analytics
BI	Business Intelligence
CPU	Central Processing Unit
DSS	Decision Support System
EBITDA	Earnings before interest, taxes, depreciation, and amortization
HDD	Hard Disk Drive
IaaS	Infrastructure-as-a-Service
IS	Information Systems
ISIN	International Security Identification Number
NSA	National Security Agency
OLAP	On-Line Analytical Processing
P/B ratio	Price-to-Book ratio
P/E ratio	Price-Earnings ratio
PaaS	Platform-as-a-Service
PO	Portfolio Optimization
RAM	Random Access Memory
SaaS	Software-as-a-Service
SMI	Service Measurement Index
TC	Transaction Costs
UGC	User-generated content
URL	Uniform Resource Locator
vCPU	Virtual Central Processing Unit
VIC	Virtual Interest Communities
VIF	Variance Inflation Factors
VM	Virtual Machine
WMW test	Wilcoxon-Mann-Whitney test
WoC	Wisdom of Crowd(s)

1 Introduction

"Nothing is more difficult, and therefore more precious, than to be able to decide."

Napoleon Bonaparte

A major task of management is taking decisions (Mintzberg 1971). While management activities range from long-term strategic planning (defining goals) over management control (accomplishing goals) to operational control of task execution (Anthony 1965), different kinds of information has to be processed for proper decision-making and problem solving. Support by information systems for these situations is equally diverse. In the year 1971, Gorry and Scott Morton coined the term *Decision Support Systems* (DSS) to allow a more specific distinction of the general term *Management Information Systems* (Gorry and Scott Morton 1971; Hosack et al. 2012). This distinction founds on a combination of the management activities above – strategic planning, management control and operational control – and Simon's classification of decision tasks (Simon 1960) into structured, semi-structured and unstructured problems. Structured (or *programmed* in Simon's terms) tasks are well-understood, routine and easily solved, while unstructured (or *nonprogrammed*) tasks are novel, difficult and without existing solution scheme. Gorry and Scott Morton thus defined *Decision Support Systems* as computer systems which support tasks that comprise at least one semi-structured or unstructured component (Shim et al. 2002; Gorry and Scott Morton 1971). Thus, they support decision makers in difficult and complex situations to increase quality and efficiency of (management) decisions. Since then, DSS have become a major area of Information Systems (IS) research with an ever rising number of publications and a dedicated journal (Hosack et al. 2012). Advances in Internet technology, analytical and data processing methods and IT infrastructure create new complexities requiring decision support, but also new opportunities to improve decision support systems and hence advance the field, as this thesis shows.

By today, DSS has become a term of its own and is also used for systems assisting in structured problem domains (Hosack et al. 2012). They help to manage complexity and assist decision makers to overcome bounded rationality and consider all important aspects of a decision (Hosack et al. 2012). The works presented in this thesis embrace such a wider definition and consider DSS as any system that contributes to the quality of a decision (e.g. recommender systems), whether it is eventually taken by humans or automatically. Especially for complex decision problems, DSS enable a

co-operative problem solving approach, combining the strenghts of humans and machines for effective solutions (cf. Schoder et al. 2014). In addition, over time, when problem domains get better understood and thus more structured, the ability to *program* solutions (if possible) increases and hence provides for more automated decision support (Hosack et al. 2012).

1.1 Current IS Developments and Thesis Positioning

This thesis examines Decision Support approaches in the light of three current major developments in the IS field which promise exciting opportunities to advance the field of DSS, especially in combination: The rise of the Web 2.0 enabling the Internet to become a social medium yielding user-generated content; the emergence of (Big) Data Analytics as the latest chapter in the area of Business Intelligence (BI) and Business Analytics (BA); and finally, the transformation of IT infrastructures by the arrival of the Cloud Computing paradigm which increases flexibility of IT infrastructures.

Those three developments complement each other from the perspective of DSS (Figure 1-1): The Web 2.0 provides rich data sources for analyses while also requiring new methodological and technical approaches due to the structure of these data which are researched within the Big Data Analytics field. These new technologies for sophisticated analyses of huge and heterogeneous data require large amounts of computing and storing resources and thus stimulate cloud computing usage (Armbrust et al. 2009). The Cloud Computing paradigm with its Infrastructure-as-a-Service (IaaS) layer (cf. Chapter 1.2.4) in turn facilitates the delivery of an appropriate infrastructure for such Big Data Analytics. By easened use and management of large scale data centers to reap economies of scale, it provides a flexible and cost-efficient way to access such resources for execution, also in variable and irregular intervals. Additionally, the Software-as-a-Service (SaaS) layer of Cloud Computing (cf. Chapter 1.2.4) is a convenient channel to provide a DSS user interface and to deliver functionality from decision support systems. The Decision Support field makes use of Data Analytics methods which fulfill the requirements of its problem tasks and deliver building blocks for solutions. Results from Decision Support disseminate into theory and practice, leading to improvements in other areas such as Social Media and Cloud Computing as they enable better decisions in those fields.

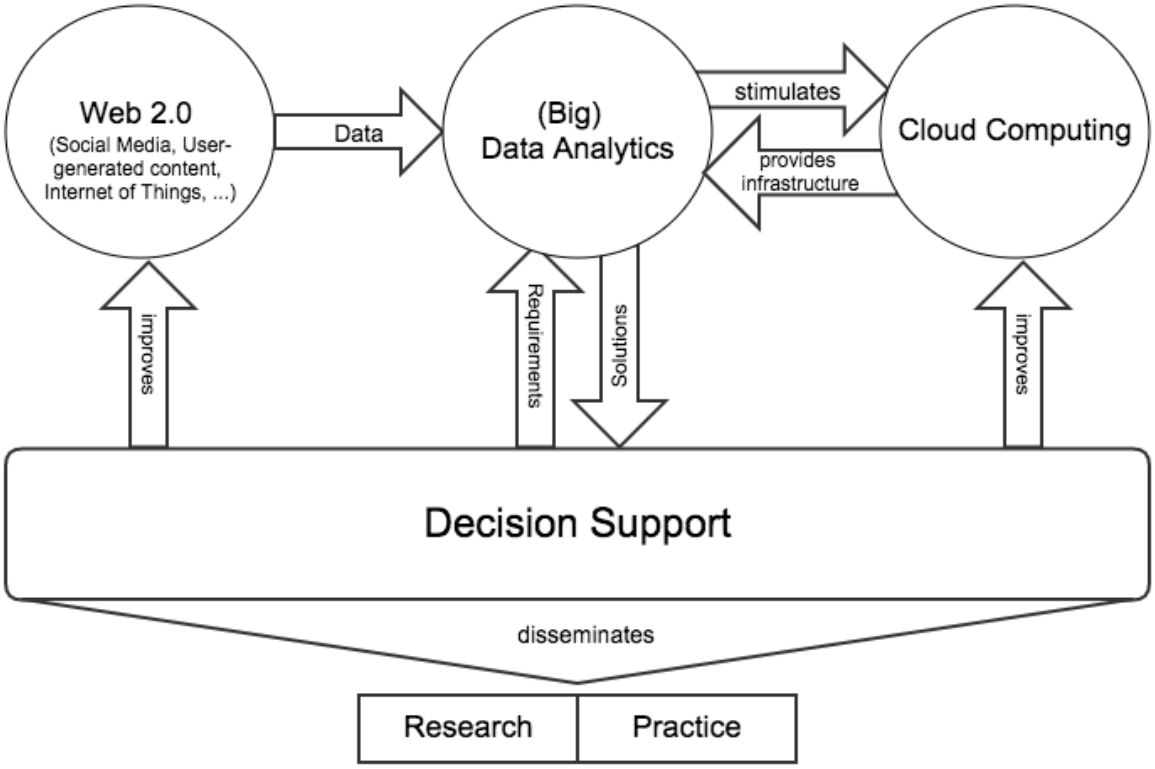

Figure 1-1. Research context of thesis

As this thesis spans a wide range of topics and applications, the following section will provide a general introduction on each of the three research contexts mentioned above with their specific relation to DSS and this thesis' articles. More detailed introductions to the specific topics are contained in the respective articles. After this introduction to the area of research, I will give an overview of the thesis structure and the individual articles.

1.2 Research context

1.2.1 Web 2.0 – User-generated content and Social Media

At the turn of the millenium, a combination of technologies enabled the World Wide Web to become an interactive medium, called Web 2.0 (DiNucci 1999). Instead of static document uploads and downloads, users were now able to easily add content to the web interactively and communicate with each other – a major enabler for the creation of social media platforms. Such *user-generated content (UGC)*, i.e. data, information, ratings or media submitted by (an unspecified group of) regular people over the Web (Krumm, Davies, and Narayanaswami 2008), led to new applications on the Internet like Wikipedia, restaurant review sites or special interest communities e.g. in the Finance area. Early forms of interactive communities on the Web like discussion

forums, chats etc. transformed with technological advance and increasing user adoption into what we call *Social Media* today with a broad spectrum of applications and providers. Among many others, we distinguish social network sites like Facebook, blogging platforms like Twitter or tumblr, but there are also examples for more specific use cases such as Flickr for picture sharing or last.fm for music interest sharing (Solis 2016).

While Shim et al. (2002) mention Web access primarily as an efficient way of user access for DSS (Shim et al. 2002), its role and meaning expanded to an important source of data by the creation of Web 2.0 and, soon, the Internet of Things[1]. Data from such social media platforms is able to serve as a data source for DSS (Hosack et al. 2012; Zhang and Yue 2014; Cortizo, Carrero, and Gómez 2011) and has been applied e.g. to detect sexual predation in online chats (McGhee et al. 2011) or to recommend websites on social bookmarking sites based on past preferences (Bogers and van den Bosch 2011).

Recommender systems as a specific kind of DSS are also able to profit from social media data, e.g. for collaborative filtering approaches based on user tags (H.-N. Kim et al. 2010) or news recommendation (Q. Li et al. 2010). Chapter 3 of this thesis presents a recommender system which uses Facebook profile data by connecting with a user's Facebook account to derive product recommendations and thus supports users in their search of interesting products. Another part of this thesis (Chapter 4) applies the data of a finance community site called *Sharewise*[2] where people exchange insights or opinions about stock performances in a structured way, assigning *buy* and *sell* recommendations and a target price to support investment decisions in the stock market.

1.2.2 Wisdom of Crowds (WoC)

UGC is suitable as a data source for research and decision making even for purposes not recognized at the time of content creation (Lukyanenko et al. 2014). One specific form of harnessing the potential of UGC for decision problems is the effect of the *Wisdom of Crowds* (Surowiecki 2005). This form of collective intelligence describes that a group of average people under certain conditions is able to achieve better results in a

[1] Internet of Things describes the transformation from an Internet of computers to a network of (small) intelligent and potentially sensing devices that blend into the user's environment to enable ubiquitous computing (Weiser 1991; Mattern and Flörkemeier 2010).

[2] http://www.sharewise.com

problem solving situation than any individual of the group (Leimeister 2010). The vast availability of UGC through the rise of social media boosted the application of crowd wisdom in decision tasks either to generate alternative solutions or evaluating alternatives (Leimeister 2010). For the Wisdom of Crowd (WoC) effect to occur, some preliminary conditions have to be fulfilled: Participants need to have some basic individual piece of knowledge about the problem domain; the motivation to participate in the search for a solution (Simmons et al. 2011); a diversity of opinions; and individual decisions have to be made independant of each other in a decentral execution of the decision process, i.e. consenus building must not occur (Leimeister 2010).

With regard to the area of finance, research studied the application of WoC to support investment decisions at the stock market. Antweiler and Frank (2004) find effects of financial discussion forums on market volatility and stock returns (Antweiler and Frank 2004). Hill and Ready-Campbell (2011) use data from CAPS, a platform supporting structured stock votes similar to the Sharewise site mentioned above and show that investment advice based on crowd evaluation – and even more by a selection of experts from a crowd – outperforms the S&P 500 stock index (Hill and Ready-Campbell 2011). Nofer and Hinz (2014) demonstrate that the crowd on the Sharewise platform is even able to outperform institutional analysts from banks or research institutions.

Based on these encouraging results, one article of this thesis develops a mechanism and implements a system to process stock vote data from such a community platform into a target layout of a stock portfolio (see Chapter 4). This enables to reap the benefits of crowd wisdom to support stock investment decisions by automating the high effort of data processing and analyses with a DSS. Building on that implementation of such a portfolio management system, we refine the possibilities of an investor to control the decision process by a sophisticated set of parameters to describe his/her investment approach. Chapter 5 extends this work by introducing a formal model to define investment strategies which we compile based on an extensive literature research for relevant determinants of portfolio performance. The portfolio management system is able to deploy and execute such a formal strategy specification on appropriate data streams (see Chapter 5).

1.2.3 (Big) Data Analytics

The availability and processing ability of large amounts of detailed data has increased in recent years fueling the advancement of data analyses technology and methods, paraphrased with terms like Data Sciene or Big Data Analytics (Agarwal and Dhar 2014). The Internet of Things is to become a ubiquitous source of massive data suita-

ble for a large spectrum of analytical purposes (cf. Riggins and Wamba 2015). As mentioned above, UGC serves as an example for a growing source of data which is readily available from the Internet and even though it was not necessarily collected for that specific purpose, it is nevertheless suitable for scientific analyses (H. Chen, Chiang, and Storey 2012; Goes 2014) and decision support. Technologies to deal with such large and unstructured data sets are crucial to enable processing and utilization of these new data sources. Thus, the field of (Big) Data Analytics research is at the very heart of decision support, as it provides methods and technologies that enable new approaches to solve decision problems. For example, parts of this thesis demonstrate how to employ Facebook profile data for product recommendations (Chapter 3) and stock ratings from a stock community platform to derive a stock portfolio (Chapter 4).

The specific challenges that come with the processing of such large and diverse data sources leads to a rising importance of data analytics and are described as the 4 V's: Volume, Velocity, Variety and Veracity (Goes 2014). The first two refer to quantity and update frequency of data streams while the latter two describe their heterogeneous nature (e.g. structured, unstructured, numerical, sounds, images etc.) and the difficulty of ensuring its validity. To apply those data for decision support, a diverse number of techniques from statistics, econometrics, machine learning, and other areas are available, but their effective application and configuration for wide-spread use in decision support is still at an early stage of research (Goes 2014).

Such analytical approaches have potential for both research and practice. For research, they serve not only as a means of hypotheses testing, but also for hypotheses generation and thus a first step towards theory building (Agarwal and Dhar 2014; Chang, Kauffman, and Kwon 2014). To advance research of analytics application in DSS, a two-pronged combination of design-oriented approaches and behavioristic research might be useful (cf. Schoder et al. 2014): Build an artifact solving a problem, then use it to further study the problem domain and the effectiveness of the solution, deriving insights for solution improvement. Within the design science approach, evaluation of solution designs and demonstration of their effectiveness has ever been a core part for validation (Hevner et al. 2004), creating the need for a solution implementation.

Also from a practical perspective, data analytics enable opportunities for decision support on new business models. The validated learning approach by Ries (2011) has become popular which defines business model development as a cycle of hypothesis formulation, testing and conclusions for the next iteration (Ries 2011). Thus, experimentation and evaluation of approaches is a fundamental task not only for research,

but also for businesses nowadays which need a fast and efficient way to decide how to adapt to a quickly evolving environment. At the same time, businesses need to ensure their on-going business activities are funded in a sustainable way. They face a dilemma of investing into future business opportunities research and earning money with the current state of their business model. March (1991) called this situation of an organization the exploration/exploitation trade-off (March 1991) (cf. Chapter 2).

Transferring this perspective to e-businesses whose major part of value creation is implemented in their IT systems, we identify the need for an appropriate IT architecture allowing flexible testing and validation of different solution approaches and a quick identification of superior performing solutions. This thesis contains a proposition for such an architecture in Chapter 2 using statistical tests to determine better approaches among a set of different solutions for a common problem. In the context of Data Analytics, such architecture allows to establish an operational approach to solve the dilemma of exploitation and exploration. Enterprises can attach new system approaches to this hub-like approach and will be supplied with test cases while performance is evaluated right from the beginning and quickly decide for the beneficial approaches to improvement.

(Big) Data Analytics thus play a fundamental part in the area of decision support: They provide the means for DSS to use any available data for the purpose of decision improvement and the methodologies to support decision problems. DSS in turn define requirements to the Analytics field to foster further research and define valuable goals.

1.2.4 Cloud Computing

Breaking its path as the IT infrastructure paradigm of the 21st century, Cloud Computing promises ubiquitous, convenient, on-demand access to a shared pool of configurable resources such as applications, services, servers, storage, network etc. (Mell and Grance 2011). In Cloud Computing, costs are usually billed according to actual usage of resources (Armbrust et al. 2009) thus transforming the fixed costs of physical hardware into variable costs of on-demand utility computing. We distinguish three service models: *Software-as-a-Service* (SaaS) where the vendor provides access to a software application for a specific business purpose e.g. travel planning; *Platform-as-a-Service* (PaaS) which provides more high-level components in a specific execution environment and tools for the development of infrastructures and applications (e.g. a managed database or an address validation service); and *Infrastructure-as-a-Service* (IaaS) which provides basic infrastructure components like servers, storage and networks. In the IaaS level, computing resources are usually software-defined shares of

server farms where a virtualization layer splits a pool of physical resources upon request into instances of virtual machines (VM) of the required size. Because VMs are virtual and resource utilization varies over time (usually below 100%), providers are able to *over-provision* resources, i.e. they sell more capacity in virtual instances than they have in physical hardware. Examples of important IaaS technology vendors are the OpenStack consortium and VMWare. Gartner projects a market growth of public IaaS cloud services of 38.4% up to a volume of $22.4 billion in 2016 (Gartner 2016).

However, while IaaS might eventually lead to a standardization of IT resources, the market offers by providers are currently difficult to compare for cloud consumers. Differences in product configurations (e.g. Repschlaeger et al. 2013) and pricing schemes (cf. El Kihal, Schlereth, and Skiera 2012) increase effort for provider selection, especially when compiling more complex IT infrastructure setups. In addition, demand profiles of consumers differ as well, increasing the effort for provider identification as no “best of all” provider can be identified. An intermediary who acts as a broker between vendor and consumer helps to reduce the effort for market research and increase the efficiency of the match process between supplier and consumer. Therefore, several approaches of cloud service brokerage or comparison have been suggested in literature (Patiniotakis, Verginadis, and Mentzas 2014; Yangui et al. 2014; e.g. Garg, Versteeg, and Buyya 2013). This thesis contains an article (Chapter 6) suggesting a combined approach of price and performance comparison. By measuring performance of computing resources and including those results in a tariff comparison next to prices, we avoid a decision based on price only which might penalize quality vendors and thus holds the threat of creating a lemon market (Akerlof 1970).

1.3 Thesis Structure and Synopsis

This thesis consists of five articles which have all been published in peer-reviewed outlets between 2013 and 2015: two journal articles (*Electronic Markets* and *Decision Support Systems*) and three conference articles (two at the European Conference on Information Systems and one at the Internation Conference on Information Systems).

Figure 1-2 shows an overview of the chapters and articles. While Article 1 introduces a fundamental IT architecture to experiment efficiently with new solution approaches, the remaining four articles demonstrate DSS solutions in the research contexts introduced before. Article 2 to 4 use Social Media data to support decision tasks: Facebook profile data to support product choice or community stock votes to support investment decisions. Article 4 complements the DSS introduced in Article 3 by a more formalized approach to set parameters for an investment strategy. Article 5 provides support for

provider choice in the field of Cloud Computing. In the following, I give a short overview of every paper. Following the introductory part, chapters 2 to 6 present the five published articles. The articles were slightly revised for language and in order to achieve a consistent layout throughout the thesis. Figure 1-2 shows an overview of the chapters and articles.

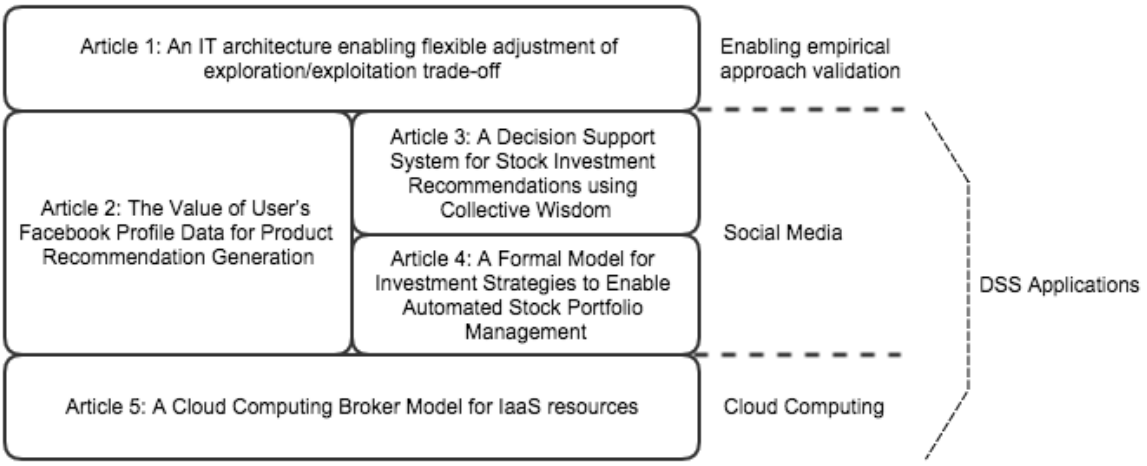

Figure 1-2. Thesis Structure

First article (Chapter 2): An IT architecture enabling flexible adjustment of exploration/exploitation trade-off

Gottschlich, Jörg (2013): An IT architecture enabling flexible adjustment of exploration/exploitation trade-off. European Conference on Information Systems (ECIS) 2013, Utrecht, Netherlands. Paper 218. VHB Ranking B.

The first article provides an architectural solution for enterprise IT systems to a fundamental problem organizations face: How to model the trade-off between exploration of new opportunities and the exploitation of valuable knowledge and established value creation processes (March 1991)? To support an appropriate long-term business strategy with an aligned IT architecture, the article suggests embedding a balancing mechanism within systems processing user requests (e.g. web applications). This balancing mechanism is attached to different solution approaches for a certain problem and dispatches user requests to each of them while monitoring a suitable performance measure for all of these approaches. It ensures that every approach receives enough test cases while determining by a frequent comparison of the approaches' performance using statistical tests if a judgment about an approaches performance is justified (=exploration phase). Once a winning approach is identified by significant test re-

sults, it receives a larger share of incoming user requests to reap the benefits of superior performance (=exploitation phase). A prototype and evaluation with encouraging results show the viability of the proposed architecture design for a product recommender scenario.

Second article (Chapter 3): The Value of User's Facebook Profile Data for Product Recommendation Generation

Heimbach, Irina / Gottschlich, Jörg / Hinz, Oliver (2015): The Value of User's Facebook Profile Data for Product Recommendation Generation. Electronic Markets, 25 (2), 125-138. VHB Ranking B.

This article is an extended version of a previous conference article[3] and examines how user profile data from a social network site like Facebook serves to derive product recommendations for a user. In a setting where no history about a user is available, this data source helps to overcome the cold start problem (Huang, Chung, and Chen 2004) of recommenders simply by connecting with a Facebook account. The article introduces and evaluates different approaches of using Facebook profile data to search for matching products in a database of app. 2 million products. Those approaches include the direct search of profile terms (especially *Likes* from profiles) within the product data as well as more sophisticated approaches like matching for brand names, identification of and search within product categories (e.g. if a user likes *golf,* search for sport products related to *golf*) or search of specific TV/movie likes only in movies products.

In a first study with 86 completes, those approaches presented ten products to a user recommended based on her/his Facebook profile against a random draw baseline. The users rated their products according to their taste (McAlexander, Schouten, and Koenig 2002) and their intention to purchase (Pereira 2000) on a 100-point Likert scale. Results show that already simple direct keyword search delivers a superior recommendation than a random draw of products. A semantically enhanced search, e.g. search for *liked* films in film products increases taste ratings by up to +20 points and purchase intention by up to +10 points.

In a second study with 38 participants, based on the results of the first study, the article differentiates profile *likes* into different categories showing that TV shows (taste +16 points) and sport teams yield best results (taste +26 points). In addition, the se-

[3] Gottschlich, Jörg / Heimbach, Irina / Hinz, Oliver (2013): "The Value of Users' Facebook Profile Data – Generating Product Recommendations for Online Social Shopping Sites", 21st ECIS conference, Utrecht, Netherlands. Paper 117

cond study looks at effects of data availability measured as profile size finding that the number of *likes* in a profile has a small positive impact on recommendation quality while other metrics (number of friends, groups or events) are not significant.

In summary, the article demonstrates different approaches on how to derive product recommendations from social media profile data and measures their impact on recommendation quality, indicating which profile data seem most promising to use for recommendation generation.

Third article (Chapter 4): A Decision Support System for Stock Investment Recommendations using Collective Wisdom

Gottschlich, Jörg / Hinz, Oliver (2014): A Decision Support System for Stock Investment Recommendations Using Collective Wisdom. Decision Support Systems, 59 (3), 52–62. VHB Ranking B.

Previous research has shown that user-generated stock votes from online communities can be valuable for investment decisions (Hill and Ready-Campbell 2011; Nofer and Hinz 2014). Building on the Wisdom of Crowd concept, aggregating several single and independent stock evaluations from community members is able to outperform a market benchmark. This article suggests a portfolio management system design which supports investors on a day-to-day basis with their stock investment decisions. The system aggregates crowd votes for many stocks, deriving a performance potential estimate that ranks stocks according to the crowd judgment. Based on this ranking, the system splits up the available capital and derives a target portfolio layout. Using a prototype, the articles demonstrates the functionality in two test runs on stock data between January 2009 and December 2010: The first run invests all capital every day in the stock with the highest crowd rating and achieves a portfolio performance of +123% (+89% after transaction costs) while the market benchmark DAX only achieves +40%. A second test run applies a Markowitz portfolio optimization on the top 10 stocks of the crowd ranking list. With a 20-day rebalancing interval, it achieves a portfolio performance of +111% (or +100% after transaction costs). A risk assessment shows that the crowd approach outperforms the market benchmark and comparable public funds in terms of absolute returns and with respect to the reward-to-variability ratio, i.e. risk-adjusted.

Fourth article (Chapter 5): A Formal Model for Investment Strategies to Enable Automated Stock Portfolio Management

Gottschlich, Jörg / Forst, Nikolas / Hinz, Oliver (2014): A Formal Model for Investment Strategies to Enable Automated Stock Portfolio Management. International Conference on Information Systems 2014, Auckland, New Zealand. VHB Ranking A.

Enhancing the previous work in the third article (Chapter 4), this article develops a formal model to specify stock investment strategies. Such a formal specification provides a structure for investors to specify and store their approaches for investments, e.g. their risk attitude, stock preference, portfolio restrictions etc. A portfolio management system as presented in Chapter 4 uses such a specification as input to execute the investors' ideas regularly on recent data or to backtest investment approaches for analysis.

In this article, based on an extensive review of investment literature, we identify determinants for portfolio performance – such as risk attitude, rebalancing interval or number of portfolio positions – and formalize them as model components. A prototype implementation used within several scenarios shows the effectiveness of each parameter included to the model. With the model developed in this article, we aim to bridge the gap between a system providing passive decision support and autonomous algorithmic trading systems by creating a „control language" for an automated portfolio management system. Such a system allows researchers and practitioners to specify, test, compare and execute investment approaches with strong automation support.

Fifth article (Chapter 6): A Cloud Computing Broker Model for IaaS resources

Gottschlich, Jörg / Hiemer, Johannes / Hinz, Oliver (2014). A Cloud Computing Broker Model for IaaS resources. European Conference on Information Systems (ECIS) 2014, Tel Aviv, Israel. Paper 562. VHB Ranking B.

Cloud Computing is a well-suited infrastructure for (big) data analytics and decision support tasks. Infrastructure-as-a-Service (IaaS), as the most flexible form of cloud computing, provides great opportunities to acquire and release computing resources as necessary. However, consumers face an increasingly opaque market due to growing number of providers and tariff options. As an approach to support consumers in their decision for the right tariff matching their needs, we suggest a broker model as an intermediate between consumer and providers. Collecting pricing and performance data from the providers, the broker allows a consumer of cloud resources to specify the quantity of resource needs and qualitative restrictions (e.g. geo location). In addition, the consumer is able to specify a load profile if his application is specifically dependent on the performance of a single component such as CPU, RAM or storage. Using the consumer's request specification, the broker queries a database with performance metrics collected by benchmarking the providers' machines and uses the results for a

tariff recommendation based on price and performance while obeying qualitative restrictions as specified. The article demonstrates the application of the broker model using a prototype and data of 14 provider tariffs.

In addition to the articles included in the thesis, the following articles were also created or published during my time as a PhD candidate which are, however, not part of the thesis:

- **Gottschlich, Jörg / Hinz, Oliver (2016):** „The Open Tariff Model – Towards Efficient Cost Comparison of Public Cloud Service“, Working Paper
- **Gottschlich, Jörg / Heimbach, Irina / Hinz, Oliver (2013):** "The Value of Users' Facebook Profile Data – Generating Product Recommendations for Online Social Shopping Sites", 21st European Conference on Information Systems (ECIS), Utrecht, Netherlands. Paper 117. *(Winner of the Claudio-Ciborra-Award 2013)*
- **Gottschlich, Jörg / Hinz, Oliver (2013):** "Der Wert von sozialen Strukturdaten aus ökonomischer Sicht", in: Anzinger, Heribert M.; Hamacher, Kay; Katzenbeisser, Stefan (Hrsg.): "Schutz genetischer, medizinischer und sozialer Daten als multidisziplinäre Aufgabe", Springer Verlag, pp. 87-95.
- **Hinz, Oliver / Gottschlich, Jörg/ Schulze, Christian (2011):** "Wie aus Ratgebern Käufer werden", Harvard Business Manager, 2011 (12), 10-12.

2 An IT architecture enabling flexible adjustment of exploration/exploitation trade-off

Title	An IT architecture enabling flexible adjustment of exploration/exploitation trade-off
Author(s)	Gottschlich, Jörg, Technische Universität Darmstadt, Germany
Published in	Proceedings of the European Conference on Information Systems (ECIS 2013), June 5-8, Amsterdam, Netherlands VHB-Ranking B

Abstract

The trade-off between exploration of new ideas and exploitation of certainties create a need for managing a balance between those two concepts within organizations. To align with an associated strategy, we suggest an IT architecture with an embedded mechanism to manage this balance when trying new approaches. A prototype and evaluation with encouraging results show the viability of the proposed architecture design for a product recommender scenario.

Keywords: IT infrastructure, exploration, exploitation, trade-off, recommendation agent

2.1 Introduction

A branch of a large international online retailer (several billion EUR annual turnover; 30,000 employees; anonymous for confidentiality reasons) operates an online shop users can subscribe to with their Facebook profile to get product recommendations matching their interests. Since all products are offered and sold by partner stores, the operator has no user purchase history available and it becomes crucial to make proper use of the user's profile data to identify suitable products for him or her. This is a challenging task and literature is scarce on the topic of which Facebook profile data is valuable for product recommendations. Hence, there is a need for experimentation to identify successful approaches in using profile data for product recommendation, as effective product recommenders lower search cost for users and enable shop owners to better satisfy customer preferences (Hinz and Eckert 2010). It is therefore important to display relevant results to users as early as possible in order to not turn them off from using the shopping site due to disappointing product suggestions.

In this setting, the company faces an instance of the classical exploration/exploitation dilemma described by March (1991). While exploration is important for being innovative to create new opportunities, exploitation plays an important role to benefit from already available knowledge. When competitiveness depends more and more on superior analytical capabilities over the competition (Davenport 2006), a high flexibility to switch back and forth between experimentation and the quick usage of identified successful approaches becomes a strategic objective (Hitt, Keats, and DeMarie 1998). Specifically, but not limited to, in an E-Business environment as described, IT support has influence on a company's success (Melville, Kraemer, and Gurbaxani 2004; Chan and Reich 2007) and need to be able to align with the strategic objectives of the business (Henderson and Venkatraman 1993).

Hence, to foster the flexibility of an IT infrastructure that aligns with a strategic objective of intense exploration/exploitation cycles, we suggest an IT architecture enabling a quick switch between exploration necessities and exploitation opportunities. In order to do so, the architecture implements a Meta-System which is connected to several subsystems ("candidate systems") – one for each approach that is to be tested ("exploration"). When in operation, the Meta-System receives a user request, selects one of the candidate systems to process the request and presents the output (like the product recommendations in the example above) to the user. Which candidate system is used to produce the output cannot be determined by the user. Feedback provided by the user (e.g. explicit feedback, click-through-rate, purchase) is used to track the performance of each candidate system. Over time, as more and more user requests are processed, the Meta-System is able to identify better performing approaches and over-

weigh them in the selection process to make use of their superior performance ("exploitation"). Thus, the Meta-System provides a controlling instance to continuously balance between exploration and exploitation without interrupting operations.

In a case study, we examine the viability of the proposed approach. A prototype implementation for the introductory product recommendation example has been implemented and user feedback data has been collected. We use the data to verify the functionality of the presented architecture. In a performance comparison, we show how such an architecture is able to create a potential benefit surplus.

The rest of the paper is organized as follows: In section 2.2, we provide the theoretical foundation for the development and understanding of our approach, followed by a methodological introduction in section 2.3. Section 2.4 introduces the architecture developed with a specific focus on the exploration/exploitation-balancing component. We show an example run in section 2.5. The paper concludes with a summary of the results and an outlook on future improvements of this approach.

2.2 Theoretical Foundation

The relationship between exploration and exploitation plays an important role in organizational development. Following March, exploration of new possibilities includes activities such as "search, variation, risk taking, experimentation, play, flexibility, discovery, innovation", while the exploitation of old certainties is characterized by terms like "refinement, choice, production, efficiency, selection, implementation, execution" (March 1991).

Striking the right balance between those two concepts is a crucial task: Focusing too much on exploitation can lead organizations to be stuck in suboptimal equilibria while engaging too much in exploration without paying attention to exploitation bears the danger of having the cost of experimentation without being able to reap the benefits (March 1991).

One aspect of the relationship between exploration and exploitation is the question of whether the two are orthogonal or continuous concepts, i.e. do they form a zero-sum game such that one can only be increased on cost of the other or can they be carried out rather independently without the need to make a trade-off? (Gupta, Smith, and Shalley 2006) The answer to this question depends, among others, on the scarcity of resources (do they compete for resources?) and the level of analysis (individual vs. complex organizations, i.e. can the tasks be spread to be carried out independently?).

In this paper, we take the perspective of exploration and exploitation being continuous concepts as we operate on a scare resource: user requests. Given the stream of incoming user requests, we need to decide if it rather serves explorative or exploiting purposes and every user request can only serve one purpose.

Another question on the interplay of exploration and exploitation is which mechanisms can be used to achieve a balance of those two concepts (Gupta, Smith, and Shalley 2006). A "synchronous pursuit of both exploration and exploitation via loosely coupled and differentiated subunits or individuals, each of which specializes in either exploration or exploitation" (Gupta, Smith, and Shalley 2006) is called "ambidexterity" (Benner and Tushman 2003). This type of pursuit can be seen as following a parallel approach as opposed to a serial pattern which is called "punctuated equilibrium" – meaning that periods of exploration are followed by periods of exploitation to form a balanced equilibrium in time (Gupta, Smith, and Shalley 2006; Burgelman 2002).

Why is the balance of exploration and exploitation important to organizations such as companies? Levinthal and March (1993) argue that "[t]he basic problem confronting an organization is to engage in sufficient exploitation to ensure its current viability and, at the same time, to devote enough energy to exploration to ensure its future viability." He and Wong (2004) found evidence that the interaction between explorative and exploitative innovation strategies has a positive effect on sales growth rate and conversely, an imbalance between those is negatively related to sales growth rate (Z.-L. He and Wong 2004). Also Kim et al. (2012) conclude that firms may emphasize one of the two concepts at any time, but that over time a balance should be maintained.

Now, seeing the right balance of both exploration and exploitation as a strategic objective, the question arises how IT systems can be aligned in support of this objective. This might especially be important for E-Businesses as their organizational structure constitutes largely of IT systems by definition, yet traditional businesses also benefit from an alignment of IT and business strategy (Chan and Reich 2007). Additionally, Ten Hagen et al. (2003) stress the importance of exploration inside recommender systems to avoid being stuck in local optima and hence they use an explorative approach to adapt to users when recommending products.

In summation, there is evidence that an architecture enabling companies to dynamically balance exploration and exploitation in IT systems is a relevant task and thus the goal of this paper is to contribute a suitable architectural design.

2.3 Methodology

Our goal is to create a system that helps to overcome the exploration/exploitation dilemma. As we want to create an artifact, we follow the Design Science paradigm which describes an approach rooted in the engineering sciences. A common methodology in this area is suggested by Hevner et al. (2004). They provide several guidelines which we follow in the construction of the proposed architecture (Hevner et al. 2004):

- **Design as an Artifact:** Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
- **Problem Relevance:** The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
- **Design Evaluation**: The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
- **Research Contributions:** Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
- **Research Rigor:** Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
- **Design as a Search Process:** The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
- **Communication of Research:** Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

The introduction and the theoretical foundation in section 2.2 show the relevance of the problem of exploration/exploitation balancing. Our proposed architecture, the design artifact, addresses this problem and provides a solution in section 2.4 which is the result of the rigorous search for a solution to the identified problem of aligning IT to support a strategy of exploration and exploitation balance. Our design is evaluated in section 2.5. In our concluding remark (section 2.6) we summarize our research contribution. To communicate our research, we present the results in this paper.

2.4 Architecture

After giving a short introduction of the implementation context, we introduce our architecture design (see Figure 2-1).

2.4.1 Context & Requirements

To get a good understanding of the operating environment of our architecture recall the introductory example at the beginning. A stream of user requests arrives at the Meta-System which has a number of candidate systems attached. These candidate systems provide different implementations for a common task. The goal of the Meta-System is to track performance of the candidate systems and strive for a desired balance of exploration and exploitation.

Considering today's common technical server setups in regard to load balancing or reverse proxy systems, these might offer a convenient implementation context for the suggested architecture – or at least provide a solid technological foundation for productive implementations in a real-world scenario.

The following section introduces each part of the architecture in detail.

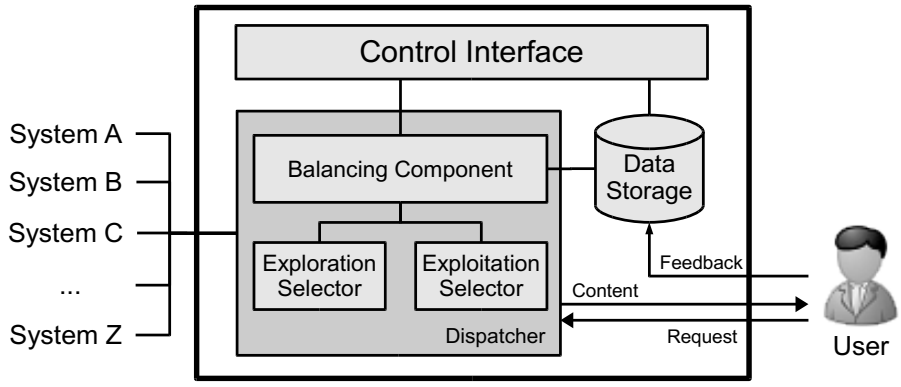

Figure 2-1. Architecture overview

2.4.2 Candidate Systems

On the input side of the system there are several candidate systems whose performance should be tested against each other. They either need to provide a uniform interface to the Meta-System or an adapter interface has to be implemented on the Meta-System side which converts the different output formats into a common structure. Please note that candidate systems may use additional data storages or remote systems but as they are perceived as closed systems by the Meta-System architecture those are not shown here.

Candidate systems do not necessarily refer to a system in the narrow sense of a dedicated server system, but can be any kind of comparable logic in a wider sense (e.g. different statistical models which are run in just one software environment or even different instances of the same system but running with different settings of performance affecting parameters).

In the example case we introduced at the beginning of this paper, the candidate systems are the different implementations for product recommenders using information from Facebook profiles (e.g. Age, Gender, Likes, Groups, Friends' Likes) to find matching products.

2.4.3 User Interface

On the output side there is a common interface to the user, usually (but not necessarily) sending his request over the internet. The Meta-System architecture provides a common interface for all candidate systems. This is important to not create artificial influences by the user interface design which could bias the user feedback from the between-subject experimental design. Additionally, the Meta-System is responsible for registering and storing user feedback.

For an example of the user output see Figure 2-2. It shows the product list that was generated based on the user's Facebook profile and provides a survey form to collect the user's feedback (in a commercial scenario one would use common performance metrics such as click-through rates, visibility time or sales). This page looks identical for every candidate system, only the selection of products changes.

2.4.4 Data Storage

A Data Storage connected to the Meta-System stores the assignment history of users and candidate system and collects user feedback for measuring the performance of each system. This data is needed in the Balancing Component to establish a favored trade-off between exploration and exploitation as shown below. In addition, the collected data can be viewed and evaluated manually via the Control Interface, not only to enable monitoring of the learning/adjustment process but also to enable control of the Balancing Component's parameterization.

The data collected in the Data Storage is also available later for additional analyses and can be seen as an asset to potentially provide further analytical insight. For the example case, the Data Storage keeps the assignment between product recommender and user, the products that are shown to each user and the survey data sent back by the user to validate the recommenders' performance.

2.4.5 Dispatcher with Balancer Component

The Dispatcher is responsible to send an incoming user request to one of the candidate systems. Its core is the balancer component, a weighted random selection process, which determines how many user requests are forwarded to a candidate system. The balancer component hence controls the frequency a candidate system is used to

serve user requests. A candidate system receives user requests to explore its performance; when the data is sufficient to make a decision, the system with the lowest performance is removed and receives no user requests anymore.

The exploration weight $w_{explore}$ is specified by the **Exploration Selector**. In principle, a simple round robin strategy would suffice to give each system an equal amount of user requests. But in a real world scenario, new systems are added at a later time, systems might have downtimes or requests are possibly processed in parallel. Therefore, a data driven approach is more reliable. Our approach computes the selection weight for a candidate system based on the difference Δ_i of test cases system i is missing compared to the system with the highest amount of test cases (Equation 1). If system A has 40 test cases, B has 30 and C 20, the selection weights for the next requests are 1 for A, 11 for B and 21 for C. The addition of 1 is needed as an initial weight and to break ties. z is the number of candidate systems available, n_i denotes the number of test cases already stored for system i.

$$\Delta_i = [\max_{j \in [1;z]} n_j] - n_i + 1 \qquad (1)$$

We normalize all the candidate system's deltas, to reach at the explore weight $w_{explore,i}$ for each candidate system i:

$$w_{explore,i} = \frac{\Delta_i}{\sum_{j \in [1;z]} \Delta j} \qquad (2)$$

Doing so, the highest weight is put on the system farthest behind in the number of test cases. When a user request comes in, a system is chosen by feeding those weights to the random selection process. The reason we still use a random selection instead of just choosing the system with the highest weight is to be less vulnerable to potential systematic biases in the experimental design.

Depending on the application scenario, other weighting mechanisms can be established. This might especially be appropriate if a new candidate system is attached next to some long running systems which have a long history of test cases. Given the max-distance approach just shown, this system would be likely to take over all user requests from the other systems. This can be intended, but attention should be paid on the desired approach to exploration weighting when new candidate systems are added.

The **Exploitation Selector** tries to identify and overweigh the most performing system(s) to quickly reap the benefits associated with the use of a high-performing system.

The basis of the performance comparison is an adequate performance metric S – e.g. click rates, Facebook "Likes" or sales (see (Page 2008) for more examples) – which will be chosen according to the individual application domain. This metric is checked frequently to evaluate performance development. In our design, the performance is re-evaluated after each "round", i.e. as soon as all systems reach a new common number of test cases (e.g. suppose every system has 4 users, when each of them reaches the 5th user, the performance of every system for users 1 to 5 is evaluated – compare x-axis in Figure 2-3).

The exact method of selecting and switching recommenders depends on the individual goal of the experiment. This paper focuses on identifying and quickly exploiting one approach out of several possible solutions and hence the overall goal is to quickly increase performance. In different settings, one might rather be interested in the actual differences of systems than a winner or loser decision. Instead of making the claim to provide a one-fits-all approach, we rather suggest that decision process provided here should be adapted to individual needs according to the specific application scenario.

In order to detect the best (or worst) performing system, we test the system with the highest (lowest) performance mean against the performance mean of all other systems within each round. If the difference is significant on a predefined level, it is possible to make the decision and either start using the top performing system exclusively or removing the worst performing system from the selection set (i.e. distribute users only to the remaining systems and continue the selection process). Executing the latter approach repeatedly, also leads to the identification of the best performing system(s) eventually – with a more precautious approach though.

Comparing only the lowest and highest mean to the mean of the remaining systems has some advantages:

- Complexity reduction: Instead of $z \cdot (z-1)/2$ tests when comparing each candidate system with each other, we only need to perform two statistical tests: one to test for a potential winner and one to test for a potential loser. Additionally, we avoid the alpha error inflation (and the associated corrections) resulting from the application of multiple tests for pairwise comparisons.
- Quick identification of distinct winners/losers: If the candidate systems' performance spread early into different levels, we can make an early decision while one-by-one comparison might lead to more indifferent findings requiring more test cases to increase test power (Bortz 2005).

To test the difference of the mean for significance, we use the Wilcoxon-Mann-Whitney test (WMW test) (Wilcoxon 1945; Mann and Whitney 1947) which is a non-parametric test and therefore doesn't put restrictions on the distribution of the user

feedback. This comes to the price of slight loss in statistical power compared to a common t-test, but increases flexibility and prevents the need and danger of making prior distribution assumptions (we rather collect a few more test cases than making an unjustified decision). However, if reasonable judgment about the distribution of the user feedback is possible, the test used can be adapted accordingly.

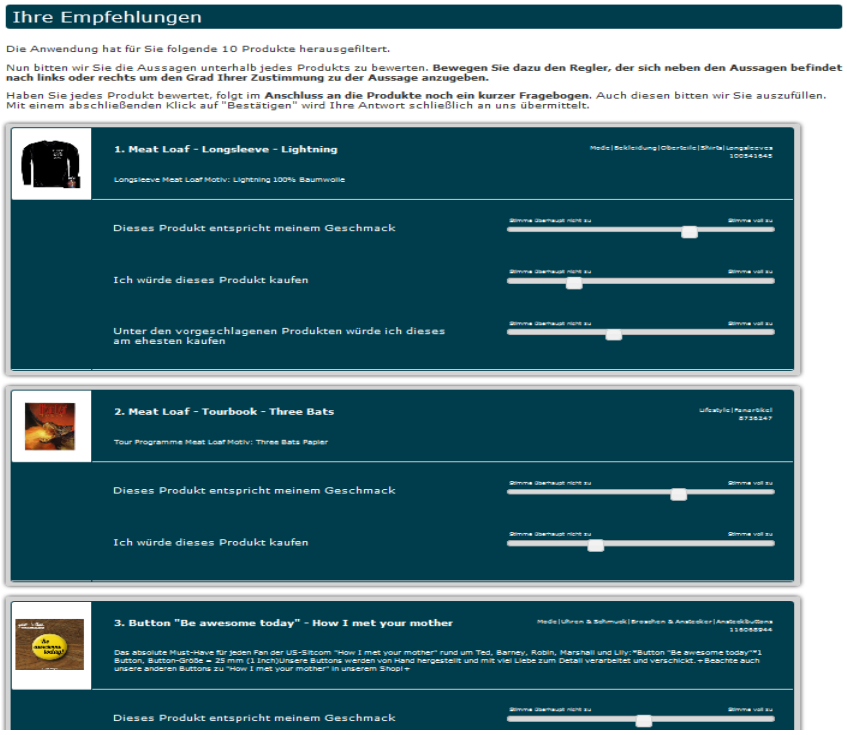

Figure 2-2. The User Interface of the Prototype presenting the result of a candidate system to the user

The null hypothesis of the test is that the two samples are from the same population of performance values. A significant test result indicates that the mean performance of the system in focus differs meaningfully from the mean performance of the alternative systems, thus the judgment over the high or low performance of the candidate system is considered as justified.

If several systems cluster at the bottom or at the top of the distribution (i.e. if their mean performance is similar) the test will not lead to a decision in the current round as, by definition, the systems' performances are considered equal. With increasing number of test cases, the performance differences can still rise to significant values in later rounds as sample size and test power increases (Bortz 2005).

In section 2.5 we present an example run which shows how the exploration decision process works over time.

2.4.6 Control Interface

In order to set the parameters of the iterative selection process (such as confidence requirements), add new candidate systems or monitor the adaption behavior of the Meta-System, a Control Interface is provided that serves administrative purposes.

2.5 Case Study

To demonstrate the functionality of the architecture presented, we are going to show a sample case in this section. It is based on the introductory example of product recommenders based on Facebook profiles. Of course, it is possible to use other social media user data to create product recommendations (and e.g. let those candidate systems compete against the Facebook-based ones), but for this prototype, we focused on using Facebook profiles only.

2.5.1 Experimental setup

We built a system that allowed users to log in with their Facebook profile and receive 10 product recommendations based on their profile from a database of roughly 2 million products. The product recommendations are presented to the user as shown in Figure 2-2. For this experiment, users were explicitly asked via questionnaire to state their satisfaction with the recommended products on 100-points-Likert scales. In a real-world scenario, the collection of user feedback would be done rather implicitly, e.g. via click-through rates, sales figures or similar measures.

The experimental run of the prototype was conducted during July/August 2012 and yielded 162 responses available for analyses. Those 162 respondents were spread among 6 candidate systems (labeled A-E and Z), yielding 27 respondents per recommender.

2.5.2 Results

Figure 2-3 shows the average user rating per candidate system. We sum up all of the single ratings of each user (maximum is 3100 points per user) and take the average rating as our performance measure for the candidate systems. It is iteratively recalculated in every round with a growing number of observations. For example, in round 10, the average is computed from 10 observations per recommender while in round 20 it is based on 20 observations, meaning the measure gets more stable over time (compare Figure 2-3).

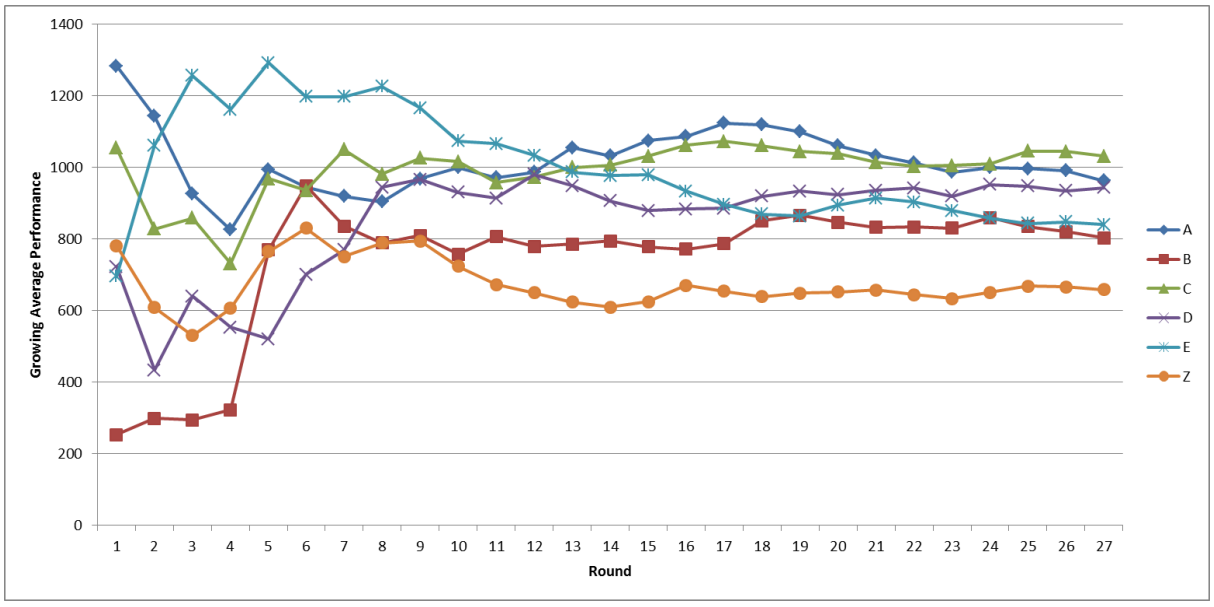

Figure 2-3. Average Performance of all candidate systems over 27 rounds

Table 2-1 shows the results of the exploitation checks. Usually the testing for exploitation possibilities starts when a predefined level of minimum observations per candidate system (depending on the specific application domain) is available. For this case, we already start right from the beginning for illustration purposes, to show how significance evolves.

For small N (i.e. in early rounds), the test results are of course unreliable. However, we see how in later rounds the test stabilizes on Z as the lowest performing system. A minimum sample size should be defined before using the test results for a selection decision. In this case Z is the most likely candidate to be removed from the set, as it stabilizes at a low performance level. If Z no longer belongs to the set of candidate systems, user requests are only distributed to the remaining five systems. By doing so, the remaining systems receive a higher amount of user requests, which accelerates the collection of further data to make a robust decision about the performance of the remaining systems.

Table 2-1. Iterative application results of the WMW test (α=5%)

Round	Lowest Performer		Common Mean of others	Wilcoxon-Mann-Whitney test			
	System	Mean		p value	conf. int. low	conf. int. high	conf. int. size
1	–	–	–	–	–	–	–
2	B	299	814	0.1212	-200	1174	1374
3	B	294	841	0.0172	91	1083	992
4	B	323	775	0.0100	84	768	684
5	D	521	957	0.1217	-50	948	998
6	D	700	971	0.2502	-271	761	1032
7	Z	750	954	0.5516	-276	628	904
8	B	788	968	0.0891	-68	767	835
9	Z	794	986	0.5221	-233	552	785
10	Z	723	955	0.3360	-151	587	738
11	Z	672	942	0.1939	-97	588	685
12	Z	649	950	0.1092	-55	602	657
13	Z	624	954	0.0527	-4	603	607
14	Z	609	942	0.0357	26	587	561
15	Z	624	948	0.0343	27	566	539
16	Z	670	947	0.0660	-20	519	539
17	Z	653	952	0.0356	24	535	511
18	Z	638	963	0.0166	66	555	489
19	Z	648	961	0.0180	58	529	471
20	Z	651	952	0.0181	46	502	456
21	Z	657	945	0.0194	43	482	439
22	Z	644	938	0.0120	73	477	404
23	Z	633	924	0.0100	70	461	391
24	Z	650	935	0.0111	62	454	392
25	Z	668	933	0.0190	39	431	392
26	Z	665	927	0.0152	43	421	378
27	Z	658	915	0.0145	43	408	365

2.5.3 Estimate of potential benefits

The prototypical character and the limited number of observations available so far are hardly a base for providing solid measures of benefit. Nevertheless, we like to provide an estimate of potential benefits which we conduct as follows:

- We assume, it is decided in round 27 to remove Z from the selection set
- We use the existing observations to simulate a second pass, but this time assigning Z the average raw performance of all other systems except Z in each round. The logic behind this assumption is that after the removal of Z the users previously sent to Z would rate the other systems similar to the users originally sent to A-E on average. So the second pass reuses the original data set, but for each round the values of Z are replaced by the average rating of its peers in that round.
- We compare the total sum of user ratings of the first pass with the second pass to derive a potential benefit from the decision to remove Z in round 27. Please note that we are using absolute user rating values here (as to measure the total gain from operating the system), while we use average rating to control the candidate system's performance in the iterative exploitation process.

The results can be found in Table 2-2.

Table 2-2. Potential benefit simulation results

	First pass (Round 1-27 as shown)	Second pass (Simulated round 28-54)	Sum of User Ratings	
			absolute	in %
Base case (Z is not removed)	141,333	141,333	282,666	100.0
Simulation (Z removed after first pass)	141,333	148,296	289,629	102.5

So under the given assumptions, we see a total surplus of 2.5% (over both passes) by the decision to switch off Z at the end of round 27. It is created by sending users not to Z anymore but rather to one of the other, better systems.

2.6 Conclusion and Outlook

In this paper, we proposed an architecture providing the capability to dynamically balance between exploring new approaches and exploiting the opportunities discovered by this exploration. We introduced und explained the architecture design and showed an example run with first available data and encouraging results. By an automated in-process control of user request distribution, the user satisfaction can potentially be increased compared to a static setup.

While the task of exploration and exploitation could theoretically be done manually, the suggested architecture enables extensive "online" experimentation without interruption of service – as the user requests are dispatched by the Meta-System, candidate systems can easily be added or removed. Additionally, concise predefined success criteria provide for a proper test setup meeting statistical requirements for a sound methodology. As a positive side effect, an operational data repository grows along with the running system which can serve additional analytical purposes (e.g. traffic or response time analyses, see also (Palmer 2002)). Thus, the application of this architecture contributes in multiple ways to support a strategy of balancing exploration and exploitation.

The product recommender case shown in this paper and implemented as a prototype is only one possible application of many. For any scenario, where a number of alternative methods exist and user feedback indicates perceived quality by the user, the architecture offers a solution to the exploration/exploitation dilemma. For example when looking at online advertising, there are different approaches of choosing the right banners for a web page visitor – e.g. based on known user characteristics or navigation behavior. Using the proposed architecture, a website operator could imple-

ment different models and let them compete against each other, using the click-through rate as a success measure.

The most important next step would be to verify the functionality of this architecture in a larger environment, e.g. within a large company, to create a bigger set of test cases, hence more stable test results and a better estimate of potential (or realized) benefits.

Even though introduced in a setting of generating product recommendations to display on a website, the approach can also be modified to suit other implementation contexts. Whenever there is a need for experimentation on a stream of user requests, an implementation should be considered. The architecture is not limited to interface with the user directly, but can also feed into another intermediate system.

3 The Value of User's Facebook Profile Data for Product Recommendation Generation

Title	The Value of User's Facebook Profile Data for Product Recommendation Generation[4]
Author(s)	Heimbach, Irina, Technische Universität Darmstadt, Germany Gottschlich, Jörg, Technische Universität Darmstadt, Germany Hinz, Oliver, Technische Universität Darmstadt, Germany
Published in	*Electronic Markets*, 25 (2), pp. 125-138 VHB-Ranking B

Abstract

Most online shops apply recommender systems, i.e. software agents that elicit the users' preferences and interests with the purpose to make product recommendations. Many of these systems suffer from the new user cold start problem which occurs when no transaction history is available for the particular new prospective buyer. External data from social networking sites, like Facebook, seem promising to overcome this problem. In this paper, we evaluate the value of Facebook profile data to create meaningful product recommendations. We find based on the outcomes of a user experiment that already simple approaches and plain profile data matching yield significant better recommendations than a pure random draw from the product data base. However, the most successful approaches use semantic categories like music/video, brands and product category information to match profile and product data. A second experiment indicates that recommendation quality seems to be stable for different profile sizes.

Keywords: Product recommendation, cold start problem, recommender, Facebook, social shopping sites

---

[4] This article is provided with kind permission from Springer Fachmedien Wiesbaden. The original version is available at: doi:10.1007/s12525-015-0187-9

3.1 Introduction

Nowadays most online shops apply *recommender systems*, i.e. software agents that elicit the users' preferences and interests with the purpose to make product recommendations (Xiao and Benbasat 2007). Recommender systems foster add-on and cross-sales and have impact on sales diversity (Hinz and Eckert 2010). They vary in the system's input, the data representation and the recommendation approach (Huang, Chung, and Chen 2004). Most recommender systems use past transactional data (e.g. on products and the user) to derive product recommendations (Adomavicius and Tuzhilin 2005). These systems, however, usually suffer from a "cold start problem", i.e. it is difficult to make recommendations for new users where no transactional data is yet available (Huang, Chung, and Chen 2004). Previous research proposed several solutions to this problem. First, a new user might get non-personalized recommendations built on top-seller rankings (Schafer, Konstan, and Riedl 2001). Jannach et al. (2010) suggest explicitly asking new users for product ratings. It might be also possible to apply user's transactional data on-the-fly, such as the navigation history in an online shop (Huang, Chung, and Chen 2004). Adomavicius and Tuzhilin (2005) suggest using external information from outside the organization's systems to build profiles of new users. A decade ago this might have been an exotic approach, but nowadays with the emergence of technologies which allow users to create and maintain online profiles, recommender designers have access to a huge body of user data.

Especially for the emerging field of social shopping sites the integration of external user data might create a promising opportunity to generate targeted product recommendations for new users. We define *social shopping sites* as online shops which integrate external online social networking sites like Facebook or offer their own features allowing users to build profiles, maintain their social relations (e.g. friendships), post their purchases on their walls or let friends evaluate their purchases. Well-known social shopping sites are caboodle.com and thisnext.com.

With respect to recommender systems, social shopping sites face the same cold start problem as conventional online shops. It might be even worse if the social shopping site is an intermediary who does not offer a product portfolio itself, but provides a market place for a high number of sellers. To this business model the cold start problem is immanent: when buyers navigate to the partner stores and make their deals there, the market platform has only limited insight into the transaction and hence can never build a detailed purchase history about its community members. However, social shopping sites have access to additional social information about the user. Based on such data, it might be possible to generate targeted product recommendations. Re-

garding the development of recommender systems this leads to an effort to incorporate social data (Bobadilla et al. 2013), which becomes more available with the further development of Internet-based services. Some approaches propose to integrate social data available within the system (e.g. J. He and Chu 2010; Y.-M. Li, Wu, and Lai 2013). To the best of our knowledge, no study has proposed and evaluated an approach based on user's profile data on social networking sites (i.e. an external data source) yet.

Because of the diversity and the vast amount of users' data on social networking sites, it is however unclear which information allows generating valuable product recommendations. Further, the users on such sites are different in terms of online activity and profile maintenance which leads to a diversity in data availability with extensive, up-to-date profiles on the one end and rather scarce profiles on the other end. Hence it is unclear how the amount of data extractable from a user's profile impacts the recommendation quality. The purpose of this paper is therefore to evaluate what kind of data on a user's social networking site profile serve as a good base for product recommendations at a social shopping site and how the degree of a profile's maintenance impacts the recommendation quality. To approach this question, we build a modular recommender system which implements different methods to generate product recommendations based on a user's Facebook profile data. Together with the social shopping site of the world's largest mail order company (anonymous for confidentiality reasons), we conducted two field experiments, asking participants to evaluate product recommendations generated on base of their Facebook profiles to measure recommendation quality. The first study investigates what kind of Facebook profile data serves as a good base for product recommendations. Based on the results from the first study, the second experiment deepens our analyses and investigates additionally whether the user profile's size has impact on the quality of product recommendations.

The remainder of this paper is structured as follows: In the following section, we embed our approach into previous research about solving the new user cold start problem. Then we present the experimental setup we used in our two studies and describe the data, i.e. the parts of the Facebook profile we used and the product database. After that, we discuss the results of our first empirical study in detail, followed by experimental details and results for the second study. Finally, we summarize our research and discuss future research opportunities.

3.2 Related Work

When a new user visits an online shop, it is important to generate reliable recommendations and build good user profiles from the very beginning (Montaner, López, and Rosa 2003) to increase the user's perceived usefulness of and trust in the recommendation system (Xiao and Benbasat 2007). As mentioned by Bobadilla et al. (2013), a new user cold start problem is one of the "great difficulties faced by recommender systems" because it could lead to a vicious circle: if a new user is not satisfied with the recommender systems at the beginning, he or she might stop using it, what in turn hinders the creation of good user profiles and hence making valuable recommendations (Bobadilla et al. 2013).

The recommender systems research community proposes several approaches to deal with the new user cold start problem. Table 3-1 summarizes these approaches suggested by previous research. These approaches can be distinguished by means of three dimensions. First, the *type of user feedback* differs between approaches: implicit feedback is generated with little user effort (e.g. re-using data created for other purposes), and explicit feedback needs an active evaluation of a proposed product recommendation by the user (Kass and Finin 1988). Second, the *source of data* can be external, obtained from outside the system, or internal, generated within the system. Third, *degree of personalization*: recommendations for new users can be personalized, i.e. customized for each user by use of individual data, or non-personalized i.e. equal for all users.

One approach to solve the cold start problem is to offer non-personalized recommendations based on top-sellers list or editors advices (Schafer, Konstan, and Riedl 2001). This approach might be insufficient for small online shops, which have the problem of sparsely rated products. Another approach is to ask new users to evaluate a selected set of products (Jannach et al. 2010). In this case the shop operators need to determine the set of products, which promise the highest information gain. Rashid et al. (2008) evaluate e.g. five strategies to select a set of items which a new user should evaluate. It might also be helpful to explicitly ask users for their interests and preferences (Rashid, Karypis, and Riedl 2008). However, the users are often not willing to provide such information (Montaner, López, and Rosa 2003). As an alternative, customer data recorded with the registration form (like name, address, age, sex) may be used to classify a new user to some stereotype to generate initial product recommendations (Montaner, López, and Rosa 2003).

Table 3-1. A taxonomy of approaches for solving the new user cold start problem

		Type of user feedback	
		Explicit	**Implicit**
Data source	**External data**	**Personalized:** Raffles, Competitions **Non-personalized:** Public statistics Market research	**Personalized:** User data from social networking platforms
	Internal data	**Personalized:** Registration form Asking for interests and preferences **Non-personalized:** Explicit rating of a selected set of products	**Personalized:** Online shop navigation history **Non-personalized:** Top-seller lists Editors' advices

Some approaches try to enrich sparsely existing data about new users. For example Kim et al. (2010) use tags provided by the users of different web sites to solve the user cold start problem (H.-N. Kim et al. 2010). Since a product can be characterized by several tags, more data is then available to generate recommendations for new users who have not bought many products yet. Weng et al. (2008) suggest using taxonomies on user's product preferences to derive, e.g. a general interest in outdoor clothes (Weng et al. 2008). Rodríguez et al. (2010) suggest a hybrid collaborative and knowledge-based system which utilizes linguistic information about the users and products (Rodríguez et al. 2010).

A recent trend in the development of recommender systems strives to incorporate social information which is increasingly available with the development of Web 2.0 (trusted users, followers, posts etc.) (Bobadilla et al. 2013). In contrast to previous recommender systems, where the similarity between users was computed by algorithms based on historic transaction data, in social network data, the users themselves provide the information about their contacts and trust relationships. For example, Li et al. (2013) suggest a recommendation system which integrates different sources of information to make a recommendation: preference similarity, trust, and user's social relations (Y.-M. Li, Wu, and Lai 2013). He and Chu (2010) suggest an approach to solve the new user cold start problem, generating product recommendations based on the preferences of users' neighborhoods in a social graph (J. He and Chu 2010). The advantage of these approaches is to receive detailed information about even new users without the need for conscious, explicit feedback by the user.

The approaches described above base on some initial information about the user available within the system. Our approach, in contrast, goes beyond this and enables generating product recommendations using external data sources for a new user for whom there is no information at all. When users log in with their social networking platform account, they enable access to users' demographics (similar to the data gained by the registration form) and users' interests and preferences (shared content,

favorites etc.). Thus, the system has access to rather rich, complete and up-to-date data which users provide voluntarily and implicitly. In contrast to He and Chu (2010) who use preferences of a user's contacts to generate recommendations, our approach tries to utilize the user's very own Facebook profile information (J. He and Chu 2010).

If we are able to use such profile data to create personalized recommendations, such an approach can help to overcome the user cold start problem simply by connecting with an already existing social network profile without the need for explicit user feedback or other external data. However, to arrive at appropriate recommendations from profile information is not straightforward. First of all, which user data needs to be applied in which way to generate valuable product recommendations? In this paper, we want to analyze how user profile data can create value for product recommendations. By doing so, we want to contribute to the existing research on solving the cold start problem and provide insights on which selection of user profile data is most valuable for product recommendation generation.

3.3 Method and Data

To gain more insights into the value of social networking sites' profile data, we implemented several recommender approaches based on different selections of profile data and conducted experiments with users to evaluate their satisfaction with the recommendations. By comparing the recommendation ratings of users for the different approaches against a random selection benchmark, we derive an indication of which profile data fields yields most value in recommendation creation. We received product data from a partner company to use in our experiments (see section 3.3.2.2). Our source of user profile data is the Facebook platform.

In our first study (see section 3.4), we use a rather broad partition of profile data to gain first insights into the value contribution of profile data (cf. Figure 3-2) and apply rather simple approaches for product identification. Based on the findings of this study, we conduct a second study, implementing a more focused approach with a fine partitioning of Like data and a more sophisticated matching mechanism (see section 3.5).

In both studies, we implement the respective recommendation logic in software and invited users online to create recommendations using their Facebook profile and rate those recommendations afterwards. The next section provides details about the experimental setup for both studies.

3.3.1 Experimental setup for both studies

We conduct a combined between- and within-subject design experiment (c.f. Hinz, Hann, and Spann 2011). This means every subject receives ten product recommendations. A subject either receives recommendations based on a random draw or on a combination of the aforementioned approaches. The evaluation of the subjects who received the random selection serves as a benchmark. As the frontend interface presents all recommendations identically (see Figure 3-1; the experiment was conducted in German), the effect on the dependent variable can be isolated to the recommendation approach and the involved profile data. Such experiments allow an identification of causal effects, which is often difficult e. g. with transactional data from the field.

Figure 3-1. Screenshot of the test system

To a large extent, participants were recruited via Facebook (as participants needed a Facebook profile to participate) by sending the URL for online participation over different channels such as personal groups or community pages. As an incentive, we raffled 10 Amazon (10 Euro each) vouchers among completed questionnaires.

The course of the experiments is as follows for a subject: The subject begins on a landing page with short instructions and starts the experiment by clicking on a start button. The subject then has to login to Facebook (if not already logged in) and authorize the Facebook application created for this experiment. If the authorization process is successful, the subject is returned to the experiment site. To create the product rec-

ommendations, the test-system calls the recommender subsystem (or the random generator) providing the Facebook access token to enable the Facebook profile access (for details on the specific recommendation logic please refer to the appropriate section of Study 1 or Study 2). The subsystem then returns a ranked set of ten product ids and the frontend presents these as recommendations to the subject (see Figure 3-1) in the specified order (best recommendation first). Each product is displayed with a title, a description, a category label, a product id and, if available, a picture. As our goal is to measure how well the products match the user's preferences, we do not provide information on prices to avoid a bias caused by different price levels.

Along with the products, a questionnaire asks the user to evaluate the product selection. For each product, the user rates whether the product meets the subject's taste (item from McAlexander, Schouten, and Koenig 2002), as well as the propensity to purchase (Pereira 2000) the product on a scale of "strongly disagree" to "strongly agree" by moving a slider. The slider's range was internally translated to a scale of 1 to 100. For control purposes, we also asked for the participant's age, gender, the experience with online shopping and their Internet usage behavior. After submission, the questionnaire cannot be changed anymore. Every Facebook profile could participate only once in the test.

3.3.2 Data description

3.3.2.1 Facebook profile data

Facebook stores an extensive amount of data about each member. A complete list can be found at (Facebook.com 2014). The availability and extent of the profile data depends on the user's attitude towards entering and making the information visible in his or her profile. Hence, we focus on a core set of profile information which is frequently available and base the product recommendations on the following subset of Facebook profile data:

- Date of birth (or age, respectively)
- Gender
- Likes: Whenever a user clicks on the "Like" button of a Facebook object (e.g. a fan page), that item is stored in the user's profile and categorized into the following categories (if applicable): Music, Movies, Television, Activities, Books, Games, Athletes, Teams, Sports, Others, Admired people
- Groups: Membership of a user in a Facebook group
- Geodata: Hometown or Current City
- Posts: Status updates posted to his/her wall by the user (free-text)

Demographic data, such as gender and age, have a long tradition in marketing research and segmentation (e.g. Zeithaml 1985; Beane and Ennis 1987) and can be used to explain differences in adoption behavior (Aral and Walker 2012), even though they might often be just substitutes for other latent factors (Fennell et al. 2003). Nevertheless, as they are readily available and deliver an outer bound for user characterization, they should be included.

Certainly, Likes are the first part of the profile data which comes to mind when thinking about user preferences as they explicitly express affinity. The same is true for membership in groups. Geodata enables to handle location-related affinities. Posts, as the least structured data available due to their free-text nature, can however provide a broad insight into the user's daily life, habits and wishes, but also require some effort to be made available for product recommendations.

3.3.2.2 Product data

We aim to use the available social data to recommend products from the product database of our business partner. The product database contains a total of 1,942,857 products in the categories Lifestyle (425,334), Fashion (954,609), Habitation (390,691) and Not Specified (172,223). Products range from physical products to services like vouchers for events. Approximately, one third of the selected products have photos attached. Table 3-2 lists the fields included in the product data.

Table 3-2. Product data fields

Field name	Description
ID	Product identifier
Title	Name of the product
Description	Textual description
Gender/age	Textual description of gender/age target group (e.g. girl, boy, women etc.)
Categories	Hierarchical product category (e.g. lifestyle, fashion and subcategories)
Brand	Manufacturer brand name
Image URL	URL of product image (if any)

3.4 Study 1

3.4.1 Product recommendation approaches

Considering the task of finding the right products for a user, the question is how we can exploit the data sources available from Facebook to arrive at an effective recommendation set. Having the data about the user is only the first step; the data's true value depends on the way how it is applied to derive product recommendations. Therefore, in our experiment, we consider different approaches of profile data application to arrive at a meaningful assessment of profile data value.

A plain and simple approach of matching products to profile entries is the direct search of profile characteristics in the product data. In the context of this paper, direct matching means that the comparison between product and profile is conducted without additional knowledge or further interpretation of the data's specific context. For example, a direct matching using Likes data simply tries to find keywords from the profile's Likes information in the product data. In detail, we implement the following approaches:

- **Likes:** Likes in a Facebook profile are categorized (see section Facebook profile data) and hence provide additional information about the kind of subject that is liked. Hence, it is reasonable to try and find products whose description matches terms that are "liked" by the user.
- **Gender/age:** Mapping the gender and age taken from the Facebook profile to products matching the appropriate segment (like girls, boys, women, men) which can in turn be used to filter products, especially exclude those not matching well (e.g. girls' toys like a Barbie puppet for a male adult).
- **Groups:** By a membership in a Facebook group, users express their association to a certain topic which might be used to identify products of interest. We use the group name to search for matching products.
- **Geodata:** Some product offerings, e.g. event vouchers or souvenir articles, are location-specific. The hometown can be used to find products that match the user's home location and enable to offer products based on regional affinity.

We refrained from doing a direct matching on user's Facebook posts, as the context and notion of keywords in posts is not unambiguous per se. More advanced approaches take the specific semantics of data fields into account, i.e. they interpret the contents and use a deeper understanding of the characteristics of product and profile data to make matches more meaningful than a simple keyword match. The specific approaches we examine here are:

- **Brand matching**: Brands offer a strong identification potential for consumers, especially in terms of communicating preferences and values (e.g. Ahearne, Bhattacharya, and Gruen 2005). Within Facebook, the brands a user relates to, can be "liked" (given the brand operates a Facebook fan page) or it can appear in the Posts or Groups sections. As our product data also provides brand names along with the products, a match based on brand names is both promising and feasible. Especially as demographic data has its weaknesses in identifying brand preference (Fennell et al. 2003, 242), this approach might overcome this shortcoming.
- **Product category matching**: Product categories provide an abstract description of the particular articles. As Facebook enables users to "like" generic terms, such as activities, a matching for product groups with the users profile entries

enables to use those abstract information for matching (e.g. if a user likes "Yoga" and there is a product category "Fashion|Sports|Yoga", it is likely that the user will appreciate products belonging to this category).

- **Video/music title matching**: Video/movie/TV show or music group names can be misleading when they are split into keywords which are then tried to match to a product (e.g. "House, MD", the TV show about a misanthropic, but ingenious doctor, is a definitive concept while a search for "house" would most likely result in many unrelated products being found). As Facebook offers specific categories (see section Facebook profile data) to show affiliation with these kinds of products, it makes sense to treat their title as an atomic term when it comes to product matching.

Figure 3-2 shows an overview of the different approaches based on the core profile data we selected for availability reasons and examples of possible resulting recommendations. A potential recommendation scenario for a female user could include trendy shoes, products related to her favorite sports (yoga), brands, baby clothes because she is a member in a respective group and fan shirts of her favorite soccer club, music band and TV sitcom. The use of Geodata might recommend a voucher for a balloon flight near the city she lives in.

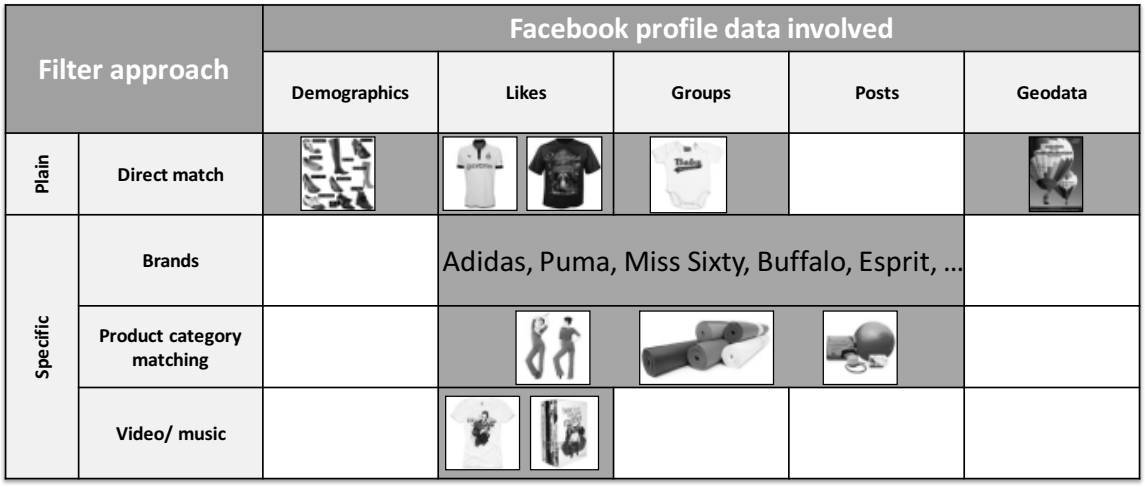

Figure 3-2. Overview of profile data used in recommendation approaches

3.4.2 Experiment

We conducted the first experiment in July and August 2012 in Germany. Three participants denied the authorization of the Facebook application explicitly (not counting those who just dropped out and closed the window without explicitly denying; this number cannot be determined with certainty). Over the course of the experiment, we collected 86 completed questionnaires rating 860 product recommendations (comprising 788 different products due to multiple selections of some of the products). 58 of

the respondents were male, 28 were female. The mean age was 27 (Std. dev. = 5.27, min = 20, max = 59, median = 26). Over 95% of the respondents use the Internet every day.

Table 3-3 shows the descriptive statistics of the respondents' evaluations of the recommended products. Every recommendation was made based on one of the approaches introduced in section 3.4.1 and thus makes use of specific Facebook data. In the following, we will examine which type of Facebook data led to the most successful recommendations. This analysis will yield first insights on the value of Facebook data for recommendation systems.

Table 3-3. Descriptive statistics of the product recommendation evaluation

Recommendation	Number of recommendations	Taste		Purchase propensity	
		Mean	Std. dev.	Mean	Std. dev.
Based on user's demographics	86	35.17	29.28	29.09	25.98
Based on user's Facebook groups	3	23.67	24.70	11.67	9.71
Based on Likes data	255	33.20	32.50	24.84	30.84
Based on geodata	5	20.20	41.82	20.80	42.05
Based on product category matching	98	36.96	31.51	29.22	30.71
Based on user's favorite brands	62	39.08	35.54	30.16	30.57
Based on user's favorite video/music	51	45.78	34.64	33.71	31.86
Random	300	23.07	27.25	19.52	26.64
Picture available	289	31.53	31.13	24.99	29.76
TOTAL	**860**	**31.35**	**31.38**	**24.75**	**29.27**

3.4.3 Model and Results

As quality metric for the recommendation we measure how well a recommendation meets the subject's taste (McAlexander, Schouten, and Koenig 2002) and the propensity to purchase (Pereira 2000), which also constitute our dependent variables. As independent variables we introduce dummy variables for the different types of Facebook data used, e.g. *Likes_D* is 1 if the recommendation is based on Likes data from Facebook, 0 otherwise or *Demographics_D* is 1 if the recommendation is based on demographic information like gender and age and 0 otherwise.

We further include demographic covariates and a dummy variable whether the recommended product included a picture because previous research found that pictures can have a significant influence on economic decisions (Dewally and Ederington 2006). We also categorized all recommended products in search goods and experience goods (Nelson 1970) manually and use this information as control variable.

Equation (1) and (2) summarize our models where *i* indicates the subject and *j* indicates the *j*-th recommendation for subject *i*:

$$taste_{i,j} = \beta_1 + \beta_2 \cdot Demographics\_D_{i,j} + \beta_3 \cdot Groups\_D_{i,j} + \beta_4 \cdot Brands\_D_{i,j} + \beta_5 \cdot Likes\_D_{i,j} + \beta_6 \cdot Geodata\_D_{i,j} + \beta_7 \cdot ProductCategory\_D_{i,j} + \beta_8 \cdot VideoMusic\_D_{i,j} + \beta_9 \cdot Pic\_available\_D_{i,j} + \beta_{10} \cdot ExperienceGood\_D_{i,j} + \beta_{11} \cdot Female\_D_i + \beta_{12} \cdot Age_i + \varepsilon_{i,j} \quad (1)$$

$$purchase\_propensity_{i,j} = \beta_1 + \beta_2 \cdot Demographics\_D_{i,j} + \beta_3 \cdot Groups\_D_{i,j} + \beta_4 \cdot Brands\_D_{i,j} + \beta_5 \cdot Likes\_D_{i,j} + \beta_6 \cdot GeoData\_D_{i,j} + \beta_7 \cdot ProductCategory\_D_{i,j} + \beta_8 \cdot VideoMusic\_D_{i,j} + \beta_9 \cdot Pic\_available\_D_{i,j} + \beta_{10} \cdot ExperienceGood\_D_{i,j} + \beta_{11} \cdot Female\_D_i + \beta_{12} \cdot Age_i + \varepsilon_{i,j} \quad (2)$$

The estimates for β_2-β_8 thus reflect the difference to the benchmark, which is a product that was randomly drawn from the product base. The variance inflation factors (VIF) are well below 4 (mean VIF = 1.13, max VIF=1.27) and thus multicollinearity does not seem to be a problem in our dataset. As each subject evaluated ten product recommendations, we have to account for unobserved factors on the individual level. As the Hausman test (Hausman 1978) showed no significant differences in coefficients between the fixed and random effects models, we decided to use random effects GLS estimation method. Additionally, we used cluster adjusted robust standard errors to account for heteroscedasticity (White-Test ($p<.05$)) and correlations within the same subject evaluations (i.e. 86 clusters). We arrive at the results for equation (1) summarized in Table 3-4.

Table 3-4. Impact of Facebook data on meeting the subject's taste.

Variable	Notation	Coef.	Cluster adj. robust std. err.
Random recommendation	*Constant*	31.13***	9.17
Based on user's demographics	*Demographics_D*	10.16*	3.08
Based on Facebook groups	*Groups_D*	16.49***	4.09
Based on Likes	*Likes_D*	10.88**	27.22
Based on geodata	*Geodata_D*	-0.423	5.32
Based on product categories	*ProductCategory_D*	15.75**	6.85
Based on brand preferences	*Brands_D*	15.85*	6.33
Based on video/music preferences	*VideoMusic_D*	19.80**	2.25
Picture available (0: no/ 1:yes)	*Pic_available_D*	1.231	0.35
Experience good (0: no/ 1:yes)	*ExperienceGood_D*	1.381	0.95
Age	*Age*	-0.384	3.45
Female (0: male/ 1:female)	*Female_D*	1.957	1.45
N = 860, Wald $\chi2(11) = 152.9$***			
+ p<0.10, * p<0.05, ** p<0.01, *** p<0.001			

The Wald χ^2 test for both models allows us to reject the null hypothesis that the variables in our models are jointly insignificant ($p<.001$ and $p<.05$), hence further analyses of the coefficients are possible. Interestingly, we find that Facebook data used for

recommendations can significantly improve the recommendation quality. The most valuable data in our context is utilized by a specific understanding of Facebook information on music, film and TV shows. Recommendations based on this information yield a +19.80 higher score on a 100-points-Likert scale ($p<.01$). A similar improvement can be made if recommendations are based on a semantic understanding of product categories that might be interesting for the subject ($p<.01$). Recommendations based on this deeper understanding of product categories receive +15.75 points on the used scale. A specific approach is also useful to identify brands and make recommendations based on this information. This yields +15.85 points with respect to the recommendation quality ($p<.05$). But even simpler approaches that use direct matching can already lead to better recommendations than a random selection, e.g. the use of demographic data ($p<.05$), Likes data ($p<.01$) or information on belonging to some groups ($p<.001$), though the latter result has to be taken with caution as the number of group-based recommendations in the data set is quite low.

Making recommendations that meet the subjects' preferences is however only an antecedent of the propensity to purchase which we analyze by estimating equation (2). Table 3-5 summarizes the estimates.

First, we observe a smaller impact of the data used with respect to purchase propensity which is not surprising: Products that meet the subjects' preferences do not necessarily convert to a purchase. This result provides some face validity.

Second, we find again that the recommendations that semantically interpret Facebook data lead to significantly better recommendations, e.g. understanding the brands preferences out of Facebook data and using this information to make recommendations, lead to an increase of +10.55 points ($p<.10$) on the 100-points-Likert scale with respect to purchase propensity. Similarly, a semantic understanding on video/music ($p<.10$) and product categories seem to be valuable ($p<.05$). Data from the video/music category can increase the purchase propensity by 10.70 points on the 100-points-Likert scale which is a promising solution for the recommender cold start problem.

Simpler approaches like the use of demographics or subject's Facebook groups can also lead to better recommendations and a higher propensity to purchase (+7.78, $p<.10$ and 8.31, $p<.01$ respectively). Although the Likes data can be useful to recommend products that match the subjects' preferences, this information delivers no surplus to the purchase propensity. This is an interesting finding, as the Like data explicitly state the subject's preferences and hence would be expected to yield higher approval.

With respect to control variables we do not find a significant impact. The purchase propensity for experience goods seems to be slightly higher as is the purchase propensity of female subjects, but these findings are not statistically significant.

Table 3-5. Impact of Facebook data on propensity to purchase

Variable	Notation	Coef.	Cluster adj. robust std. err.
Random recommendation	*Constant*	16.63^{+}	9.93
Based on user's demographics	*Demographics_D*	7.78^{+}	4.59
Based on Facebook groups	*Groups_D*	8.31**	3.00
Based on Likes	*Likes_D*	5.66	4.08
Based on Geodata	*GeoData_D*	0.16	23.86
Based on product categories	*ProductCategory_D*	10.24*	5.03
Based on brand preferences	*Brands_D*	10.55^{+}	5.72
Based on video/music preferences	*VideoMusic_D*	10.70^{+}	6.24
Picture available (0: no/ 1:yes)	*Pic_available_D*	1.40	2.30
Experience good (0: no/ 1:yes)	*ExperienceGood_D*	1.16	1.28
Age	*Age*	0.04	0.38
Female (0: male/ 1:female)	*Female_D*	1.94	3.37
N = 860, Wald $\chi^2(11) = 21.24^{*}$ $^{+}$ p<0.10, * p<0.05, ** p<0.01, *** p<0.001			

From Study 1, we learn that the use of profile data for product recommendations shows a promising impact on user's taste and, to a lesser extent, propensity to purchase. Like data is more effective for recommendation generation when we use it semantically, i.e. take the category of a "Like" into account which makes sense, as it allows matching Likes and products more targeted. This encourages further research on more sophisticated approaches how to use Like data effectively for recommendation generation.

3.5 Study 2

3.5.1 Motivation and experimental setup

The previous experimental study shows that a categorized approach of using Like data which considers the category of a Like (movie, TV show etc.) creates more successful recommendations. Hence, for our second study, we refine the system implementation and include finer-grained approaches which utilize Facebook Likes. In the first study, one approach utilized data related to entertainment (TV, movies, bands). Now we split this approach further into five modules which utilize user's favorite books, movies, TV shows, bands, and games.

During Study 1, we also noted that the extent of Facebook profile information differs between users. Some users are very active on Facebook and have hundreds of friends,

share more content frequently and also receive shared content from their friends. So we expect that the different sizes of Facebook profiles impact recommendation quality: the more information is available on the profile site, the better are the recommendations in terms of matching user's taste and increasing purchase intention.

So we conduct the second study, a) to provide a more detailed view on which type of Like fields allows generating reliable recommendations and b) to analyze how the availability of the Facebook profile data affect the recommendation quality.

3.5.2 Experiment

Based on the experiences of Study 1, we implemented a new recommender system design which focuses on exploiting Likes data from specific categories to retrieve products of those categories. The system also scans the users' status updates for relevant key words to recommend products. The re-implemented system uses a modular approach to match profile and product data and facilitates an internal scoring mechanism to create a ranked list of product recommendations. Using the new implementation of the recommender system, we conducted another experiment using a setup as outlined in section 3.3.1. Every participant received again 10 product recommendations. If the system is not able to generate 10 recommendations based on profile data (e.g. if the profile does not provide sufficient data), the system adds random recommendations to ensure at least 10 recommendations are returned. For each recommendation, the recommender stores the data field which led to the product being selected. In order to analyze the effect of user profile size and structure on recommendation quality, we collected the count of Likes, friends, groups and events for each user profile.

3.5.3 New recommendation approach

For this second experiment, we implemented a new recommendation process. The basic idea is to extract keywords (tags) from the user's profile data and from the product data and then match those two sets to identify products relevant to the user. To make matching with the product data operational, we created an index of the relevant product data fields upfront using Apache Lucene[5], an open-source search engine. This index only needs to be updated when new product data arrives. As we use a static product dataset for our experiment, we only create the index once before the experiment.

[5] http://lucene.apache.org/core/index.html

The recommendation process starts with the extraction of tags from the user's Facebook profile (i.e. Like fields and status messages). The system cleans the extracted tag set (e.g. removes duplicates and stop-words[6]). The next step uses a semantic dictionary to merge synonyms to one final tag and assigns a relative weight to each of those tags based on their occurrence frequency. Subsequently, the system queries the product index for those products and computes a similarity metric between the extracted profile data and the product data which serves combined with the tag weight as a ranking metric for the product result list. A post-processing step merges product duplicates (which occur when different modules select the same product) by adding up the different weights for one product and hence increase the product's ranking position because when a product is chosen by different modules, it seems to be more relevant to the user's preferences.

3.5.4 Results

Using this setup, we want to analyze the relative strength of the Like categories for product recommendations while considering the amount of information provided by a Facebook profile. We used the experimental setup as described before and conducted the experiment in July 2013 among German students. Surprisingly, as compared to Study 1, we had a large number of drop-outs who explicitly denied Facebook access authorization. 47 probands stopped the test right at the beginning due to explicit denial of authorization compared to three explicit denials in the first study. This huge increase might be accounted to a modification of the authorization dialog by Facebook between Study 1 and Study 2. While in Study 1, the information about granted permissions was only a small part in a dialog providing a lot of general information about the requesting Facebook application, the new dialog shows an explicit list of permissions to the user and asks to grant those to the application. Apparently, this direct question leads to a more explicit decision by the user. Another reason for more drop-outs could be the simultaneous rise of the NSA leak affair between Study 1 and Study 2, possibly increasing the users' awareness for data privacy. For an extended discussion on the issue of user privacy concerns in E-Commerce see e.g. Spiekermann et al. (2001). The data set contains 38 completed questionnaires rating 380 product recommendations.

[6] Stop words are common words defined as irrelevant for a search (e.g. "a", "and", "the" etc.), cf. e.g. (Jannach et al. 2010, 56)

Table 3-6. Descriptive statistics of study 2

Recommendation	Number of recommendations	Taste		Propensity to purchase	
		Mean	Std. dev.	Mean	Std. dev.
Based on favorite athletes	18	51.72	34.09	41.72	33.72
Based on favorite books	4	22.75	17.17	13.75	16.28
Based on favorite clothes	17	42.76	31.41	39.65	32.24
Based on favorite games	2	100	0	92	11.31
Based on favorite bands	36	35.97	32.69	18.56	22.55
Based on favorite sports teams	12	62.75	25.42	62.75	25.42
Based on favorite movies	6	39.5	24.53	26.83	24.41
Based on favorite TV shows	19	42.21	31.47	29	30.06
Based on Facebook status post	228	29.71	31.03	21.61	25.97
Random recommendation	41	23.88	28.76	17.73	22.41
Picture available	110	31.32	31.64	21.15	26.80
TOTAL	**380**	**33.13**	**31.86**	**23.87**	**27.35**

The quality metrics of recommendations are the same as in the first study: how well a recommendation meets the subject's taste (McAlexander, Schouten, and Koenig 2002) and the propensity to purchase (Pereira 2000). As independent variables we introduce dummy variables for the different more specific types of Facebook data used, e.g. *Athletes_D* is 1 if the recommendation is based on the user's favorite athlete. We include covariates for user's demographics and a dummy variable whether the recommended product included a photo.

To measure the profile size or data availability of a user, we include four variables: A user's total number of Likes, the total number of friends, the total number of groups a user belongs to and the number of events in which a user participates. Randomly generated recommendations serve as the reference category. Table 3-6 and Table 3-7. provide descriptive statistics for the second study. 79% of participants were male and the average age is 24 years. An average user has 72 Likes, 294 Friends, belongs to 13 different groups and takes part in about 2 events. For about 30% of the recommendations a picture of the product was available. 60% of the recommendations were generated on the user's status data. Other recommender's sub-modules and the random draw account for the generation of the remaining 10% of recommendations. Unfortunately, there was an unsufficient amount of recommendations based on users' favorite books, games, and movies (see Table 3-6). We exclude these observations from our further analyses. The resulting final data set contains 368 product recommendations.

Table 3-7. Users' descriptive statistics of study 2

Variable	Mean	Std. dev.	Min	25% percentile	50% percentile	75% percentile	Max
Male	.79	.41	0	-	-	-	1
Age	24.05	3.11	17	22	25	25	31
N Likes	72.21	68.22	6	24	41	104	306
N Groups	13.74	7.83	1	9	12	17	36
N Friends	294.47	100.05	106	222	291.5	355	479
N Events	2.45	2.76	0	1	2	3	13

Equations (3) and (4) summarize our models where *i* indicates the subject and *j* indicates the *j*-th recommendation for subject *i*:

$$\begin{aligned} &taste_{i,j} = \\ &\beta_1 + \beta_2 \cdot Athletes\_D_{i,j} + \beta_3 \cdot Clothes\_D_{i,j} + \beta_4 \cdot Bands\_D_{i,j} + \beta_5 \cdot Teams\_D_{i,j} + \beta_6 \cdot Shows\_D_{i,j} \\ &+ \beta_7 \cdot Status\_D_{i,j} + \beta_8 \cdot Pic\_available\_D_{i,j} + \beta_9 \cdot Male\_D_i + \beta_{10} \cdot Age_i + \beta_{11} \cdot N\_Likes_i \\ &+ \beta_{12} \cdot N\_Groups_i + \beta_{13} \cdot N\_Friends_i + \beta_{14} \cdot N\_Events_i + \varepsilon_{i,j} \end{aligned} \quad (3)$$

$$\begin{aligned} &purchase\_propensity_{i,j} = \\ &\beta_1 + \beta_2 \cdot Athletes\_D_{i,j} + \beta_3 \cdot Clothes\_D_{i,j} + \beta_4 \cdot Bands\_D_{i,j} + \beta_5 \cdot Teams\_D_{i,j} + \beta_6 \cdot Shows\_D_{i,j} \\ &+ \beta_7 \cdot Status\_D_{i,j} + \beta_8 \cdot Pic\_available\_D_{i,j} + \beta_9 \cdot Male\_D_i + \beta_{10} \cdot Age_i + \beta_{11} \cdot N\_Likes_i \\ &+ \beta_{12} \cdot N\_Groups_i + \beta_{13} \cdot N\_Friends_i + \beta_{14} \cdot N\_Events_i + \varepsilon_{i,j} \end{aligned} \quad (4)$$

The estimates for β_2-β_7 thus reflect the difference to the reference category, i.e. a product generated by the default setting of the recommender (random draw from the database). As we again used the combined between-within subject design we have to control for unobserved effects due to multiple evaluations by the same subject. As the Hausman (Hausman 1978) test showed again no significant differences between the fixed effects and random effects models, we decided in favor of the more efficient of both (i.e. random effects). Additionally, to account for heteroscedasticity (Breusch-Pagan / Cook-Weisberg test ($p<.05$)) and correlations within the evaluations of the same subject (i.e. 38 clusters) we estimate our model with cluster-adjusted robust standard errors. Table 3-8 and Table 3-9 present the estimation results. The Wald χ^2 for both models allow us to reject the null hypothesis that the sets of coefficients are jointly insignificant ($p<.001$).

The results show that the user's demographics and picture availability for products have no significant impact on recommendation quality. Some amount of variation in recommendation quality can be explained by using different fields of user's Facebook profile data. In the entertainment sector, a user's favorite TV shows allow generating high quality recommendations in terms of meeting user's taste and increase purchase

propensity. Recommendations based on these data receive +16 higher score (p<.01) on meeting user's taste but have no significant effect on user's buying decision.

Table 3-8. Results for user's taste in Study 2

Variable	Notation	Coef.	Cluster adj. robust std. err.
Random recommendation	*Constant*	29.17	19.15
Based on favorite athletes	*Athletes_D*	18.41	14.96
Based on favorite clothes	*Clothes_D*	10.30	11.04
Based on favorite bands	*Bands_D*	8.89	7.94
Based on favorite sports teams	*Teams_D*	26.28**	9.14
Based on favorite TV shows	*Shows_D*	16.18+	8.98
Based on Facebook status post	*Status_D*	-0.57	6.83
Picture available	*Pic_available_D*	-4.01	3.66
Age	*Age*	-0.16	0.67
Male user	*Male_D*	6.34	5.01
N Likes	*N_Likes*	0.09***	0.02
N Groups	*N_Groups*	-0.75*	0.34
N Friends	*N_Friends*	0.012	0.05
N Events	*N_Events*	0.014	0.54
N = 368, Wald $\chi^2(13)$= 65.06 ***			
+ p<0.10, * p<0.05, ** p<0.01, *** p<0.001			

Further, recommendations based on user's favorite sports teams yield a +26 higher score (p<.01) and +16 higher score (p<.1) on propensity to purchase. This finding might be explained by the fact that products in the product database could be better matched with information on sports. For example, if a user marks the famous German soccer team Bayern Munich as his or her favorite sports team, the system is able to find very specific merchandise products which are likely appreciated by the user. The remaining recommender sub-modules which utilize other data from a user's Facebook profile do not show significant effects on the user's evaluation.

Table 3-9. Results on user's propensity to purchase in Study 2

Variable	Notation	Coef.	Cluster adj. robust std. err.
Random recommendation	*Constant*	6.15	17.75
Based on favorite athletes	*Athletes_D*	18.83	13.94
Based on favorite clothes	*Clothes_D*	13.51	9.49
Based on favorite bands	*Bands_D*	2.87	5.77
Based on favorite sports teams	*Teams_D*	16.08+	9.39
Based on favorite TV shows	*Shows_D*	10.76	7.19
Based on Facebook status post	*Status_D*	-0.01	4.54
Picture available	*Pic_available_D*	-4.88	2.97
Age	*Age*	0.48	0.65
Male user	*Male_D*	3.50	6.03
N Likes	*N_Likes*	0.08***	0.02
N groups	*N_Groups*	-0.63*	0.26
N friends	*N_Friends*	0.02	0.02
N events	*N_Events*	-0.55	0.59
N = 368, Wald $\chi^2(13)$ =46.12***			
+ p<0.10, * p<0.05, ** p<0.01, *** p<0.001			

The next goal of the second study is to analyze how the user's Facebook profile size relates to the recommendation quality. The results show that the numbers of Likes – although small in magnitude – have a significant positive effect on both recommenda-

tion quality measures ($p<.001$). Contradictory to expectations, the effect of the number of groups the user belongs to is negative ($p<.05$). Our assumption was that the more information is contained in Facebook profiles, the better the recommendations are. The results, however, show that the effects of the data sources differ. The information within Likes leads to a better match of the user's characteristics and preferences. The information related to the groups the user belongs to worsens recommendation quality possibly because they increase the heterogeneity of the profile and hence make it hard for the recommender to identify strong preferences. The number of friends and events is not significant for recommendation quality which makes sense as our recommendation approach does not include friend or event data. So our model is not able to estimate their general applicability as an indicator of profile data availability.

Based on our data, we find that a fine-granular differentiation of Likes which allows a targeted match is useful to create successful recommendations. However, some Likes data seem to be better suitable for high recommendation quality, in our case: TV shows and sport teams. This effect might vary with the nature of the product data and the popularity of Facebook Likes categories.

Further, we could not identify strong effects of profile size in terms of number of Likes, friends, groups and events a user collects on his profile on recommendation quality. A possible explanation is that more data are not necessarily more valuable for recommendation generation. If a user "likes" things inflationary, the affection to things she or he likes might be less than for another user who chooses very carefully what she or he "likes" and hence show a stronger commitment with those statements. Another explanation, why we couldn't identify strong effects is that the profiles in our experiment all contained just enough information for good recommendations and hence additional data could not yield distinctly better recommendations. Future research should apply more detailed metrics of profile characteristics to enable deeper analyses of profile data availability on recommendation quality.

3.6 Conclusion and Further Research

As some online retailers do not have access to transaction histories and face a cold start problem when generating product recommendations, the exploitation of Facebook data to gain some first insights on the prospective buyer seems promising. We therefore conducted two studies to assess the value of Facebook data for product recommendations. As the first experiment showed, we were able to determine causal effects of the Facebook profile data on recommendation quality. With the second study,

we provided a more detailed look into the effects of different Facebook Like categories and made a first approach to measure effects of profile data availability on recommendation quality.

Interestingly, we find in our first study that the data in the Facebook profile is of value for product recommendation. Even very simple approaches like direct matching keywords from Facebook profiles with the product database lead to recommendations that match the prospective buyer's preferences significantly better than neglecting this information. User evaluations between recommendations spread though, emphasizing that developers and business practitioners have to take a close look when they want to make proper use of Facebook data. Approaches that try to interpret the data semantically and try to understand the specific meaning of the Facebook data seem to be very promising. We find that such information can increase the user's taste ratings by more than +26 scores on a 100-point scale which is a large improvement and, therefore, seems to be a promising solution for typical recommender cold start problems.

The experiment of the second study rests upon a new recommender design that enables a generic semantic approach to the use of profile data. Based on this design, we conduct a more detailed analysis of the different Like categories and confirm the usefulness of this information. The results indicate again that the performance of recommendations differs between Like categories. Developers of product recommenders based on Facebook profile data need to carefully analyze the match mechanics for their type of product and chose appropriate Like categories wisely and with respect to the nature of product data involved. Regarding the question if recommender performance depends on the user profile's data availability, the results show only minor differences in recommendation quality for different profile sizes. As we received only very few recommendations based on groups and geodata, we have only limited insight in the quality of recommendations based on these facts.

Our studies do not come without limitations: first, the entry decision in our context is not totally comparable to the real decision. In the experiments, the subjects indeed participated voluntarily but may refrain from using the system in a real setting. It is therefore not clear whether this setting led to a self-selection bias and we cannot make any conclusions whether such systems would be accepted by prospective users. The social shopping site that provided the data, however, offers a similar Facebook app and found a substantial numbers of users in the market. We conducted our studies only with German-speaking users on the Facebook platform and the results are based on this population. The language processing of profile and product data was hence based on German language processors (stop word list, semantic dictionary, lexical analysis), but all these techniques are also available for other languages, too.

However, we did not base our experiments on any explicit German or Facebook characteristics and hence see no reasons which would limit the results to a specific (offline or online) geography.

Second, beside the data quality the recommendation process also impacts the recommendation quality. Therefore, we recommend being careful when looking at the magnitude of the particular coefficients. We used a random selection as benchmark which has the advantage of an absolute, well-defined and reproducible baseline for performance comparisons. However, for a comparison with other recommendation approaches, e.g. for strategic business decisions, further evaluation of absolute performance in the specific context should be conducted. We are, however, confident that the sign and significance of the estimated coefficients are reliable evidences for the value of Facebook data for product recommendations. Our generic recommender approach in Study 2 can serve as a basis for further improvements, to enable a "soft" matching between profile and product attributes. Further work here should focus on identifying a reliable weighing model to project the preference ranking taken from the user profile into a respective product ranking. Second, for free text preference extraction, sentiment needs to be taken into account to grasp the difference if the user has a positive or negative attitude e.g. towards a brand he or she mentions on his or her profile.

This paper contributes to research on recommender systems. As our results show the value of external profile data from social networks and can be used as basis for designing recommender systems. Our work delivers starting points for developing alternative approaches for solving the cold start problem using external user data. Finally, we systematically evaluate different sources of user data with respect to their usefulness for product recommendations and give indications about the most effective selection of profile data for this purpose.

4 A Decision Support System for Stock Investment Recommendations using Collective Wisdom

Title	A Decision Support System for Stock Investment Recommendations using Collective Wisdom[7]
Author(s)	Gottschlich, Jörg, Technische Universität Darmstadt, Germany Hinz, Oliver, Technische Universität Darmstadt, Germany
Published in	*Decision Support Systems*, 59 (3), pp. 52–62 VHB-Ranking B

Abstract

Previous research has shown that user-generated stock votes from online communities can be valuable for investment decisions. However, to support investors on a day-to-day basis, there is a need for an efficient support system to facilitate the use of the data and to transform crowd votes into actionable investment opportunities. We propose a decision support system (DSS) design that enables investors to include the crowd's recommendations in their investment decisions and use it to manage a portfolio. A prototype with two test scenarios shows the potential of the system as the portfolios recommended by the system clearly outperform the market benchmark and comparable public funds in the observation period in terms of absolute returns and with respect to the Reward-to-Variability-Ratio.

Keywords: Wisdom of crowds, Investment decision support system, Virtual investment communities, Portfolio creation

[7] This article is provided with kind permission from Elsevier. The original version is available at: doi:10.1016/j.dss.2013.10.005

4.1 Introduction and Motivation

The rise of user-generated content on the Internet enabled a wider public to participate in online content creation and publication without the need for deep technical expertise. The technical possibility to centrally aggregate the local contributions of a large crowd enables the creation of artifacts which are of equal or superior quality than those made by experts in the domain. Wikipedia, as an example, reaches a comparable quality to the renowned Britannica (Giles 2005), solely depending on the contributions of a diverse anonymous crowd. This effect, coined as "Wisdom of Crowds" (Surowiecki 2005), is based on the diversity in information possession and processing of the individual members and is evident in a number of problem solving situations such as judging, estimating or decision making (Lorge et al. 1958; Forsythe et al. 1992).

Estimation tasks are a problem class where group judgments prove to perform extraordinarily well. The reason behind is an effect called *bracketing* (Soll and Larrick 2009, 782), which refers to the high likeliness that a part of the crowd will overestimate, while another part will underestimate the true value. Hence, averaging all judgments will lead to a more accurate judgment than that of the average judge (Larrick and Soll 2006). Take an example: two people estimate the outside temperature for the next day as 60°F and 80°F, while the true temperature will be 73°F. The estimates were wrong by 13°F and 7°F, or 10°F on average. However, the mean of the two estimates, 60°F and 80°F, which is 70°F, is off by only 3°F. So using deviation as a measure, the average *judgment* outperforms the average *judge* (cf. Larrick and Soll 2006).

One form of harnessing the crowd wisdom to improve decision making is the application of prediction markets. At prediction markets, participants can buy or sell contracts whose payoff is connected to a certain future event, e.g. "Candidate A will win the election". By dealing contracts over time, the contracts' prices reflect the market participants' collective judgment of the likeliness that the associated event will become true. The collective judgment has been proven to be quite close to the final result (Forsythe et al. 1992; Spann and Skiera 2003; Arrow et al. 2008). If designed appropriately, such prediction markets can be utilized to support decisions (Berg and Rietz 2003). Preference markets are a closely related concept and have been used to apply the wisdom of crowd to evaluate emerging technologies at an early stage of product development (L. Chen et al. 2009; Dahan, Soukhoroukova, and Spann 2010). The difficulty here is to prioritize resources for technologies who are most promising and which might emerge into product features. This problem is an instance of a typical investment problem and hence very close to the task in focus of this paper, the

beneficial allocation of capital to capital market shares. Chen et al. (L. Chen et al. 2009) used a preference market to compare the crowd estimate of product feature ranking to a benchmark ranking done by an expert group and found indications that they reach comparable results if the preference market provides an sufficient (non-monetary) incentive. In the perspective of the financial domain, this finding suggests that trading strategies based on crowd recommendations might be able to perform as well as public funds managed by experienced domain experts even without a direct monetary reward for the crowd members. Specifically, virtual investing communities (VICs) (see section 4.1.1) which collect user opinions on stock development, usually provide non-monetary incentives for participation such as public reputation or access to exclusive information.

4.1.1 Wisdom of Crowd in Finance

With respect to the financial domain, there is extensive research on the value of user-generated content for stock investment decisions. Antweiler & Frank (2004) analyze the information content of stock discussion boards and find evidence, that message posts can be used to predict stock market trading volume and volatility and – to a small extent – stock returns. Mood analysis from Twitter messages can be used to improve prediction accuracy of the Dow Jones Industrial Average Index (Bollen, Mao, and Zeng 2011).

Stock discussion boards evolved into Internet portals, so called virtual investing communities (VICs), where members are able to provide their guess about a share's future performance in a structured way (see Figure 4-1 for an example). Members can make a buy or sell recommendation for any share along with a target price at a specific future date (=*vote*). By doing so, the individual investors act like professional analysts and the aggregation of the single votes of a share leads to a collective judgment of its prospects. Examples for such websites are CAPS[8] or Sharewise[9]. As participants make very specific (price) predictions of a share, those platforms can be seen as a prediction market for share prices and are as such an enhancement of stock discussion boards with unstructured free-text information (Avery, Chevalier, and Zeckhauser 2011) that has to be preprocessed for analysis (including a certain loss of accuracy).

There is evidence in literature that information from these stock communities can be used to implement profitable stock investment strategies. Using data of the CAPS plat-

[8] http://caps.fool.com/

[9] http://www.sharewise.com/

form, Avery et al. (2011) find that stocks ranked highest by the community indeed show a better subsequent performance than those that were ranked low (Avery, Chevalier, and Zeckhauser 2011). Especially short (i.e. sell) recommendations of the crowd are able to predict stock price declines. Further, they analyze the composition of the performance and find that the advantage of the crowd comes from stock selection rather than market timing or style/risk factors (as identified by Fama & French (Fama and French 1993) and Carhart (Carhart 1997) which classify stocks by their market or risk profile).

An in-depth analysis of the CAPS data for investment purposes can be found at Hill & Ready-Campbell (2011). They find evidence, too, that a portfolio based on crowd voting is able to outperform the market index (S&P500) and that the higher rated shares do indeed perform better. Specifically, they find that a crowd of about 250 people always outperforms the S&P 500 index. In addition, they rank the users according to their past performance and find that a selected group of experts from the crowd performs better than the whole crowd. They test several investment strategies for portfolio construction, but disregard transaction cost. While their analysis is comprehensive and their results are insightful, investors who want to make use of the effects for future scenarios are left with extensive analytical effort and not much guidance on how to transform results into actionable investment decisions.

The same holds for Nofer & Hinz (2013) who empirically show that the average professional expert from the financial service industry and the average private crowd member are able to outperform the market. More surprisingly they also show that investors are on average significantly better off when trusting a crowd recommendation than following the advice of professional experts from banks. However, the authors do not provide and evaluate a system that implements their findings and which offers decision support for investors (Nofer and Hinz 2014).

Making use of the value in crowd data for day-to-day investment decision is not a trivial task. Because analyses are complex (both in terms of method and data) and time-consuming when done manually, a decision support system proves to be helpful.

4.1.2 DSS for Investment Decisions

There is a vast amount of literature about systems designed to support stock investment decisions with a large diversity in focus and approach. One stream focuses on asset and liability management (ALM) topics suited for professional institutions like banks which seek support for risk management comprising all of their asset classes. Moynihan et al. (2002) suggest a DSS that forecasts the amount of assets and liabilities and the primary interest rate and involve simulation models to conduct gap analy-

sis and rate risk of the institution. Additionally, it is possible to run "what-if" scenarios to analyze developments under changing market conditions (Moynihan et al. 2002). A more recent approach utilizes complex stochastic programming methods to support optimal strategic asset allocation providing a user-friendly web interface (Beraldi, Violi, and Simone 2011).

In regards to the topic of stock investment, the majority of previous research strives to provide better insights to investors by improved information support. Methods to model stock price development using optimization or machine-learning approaches are commonly used. Specifically, artificial neural networks show broad coverage in investment decision support literature, often in combination with other approaches. Tsaih et al. (1998) combine a neural network approach with a static rule base to predict the direction of daily price changes in the S&P 500 stock index futures which outperforms a passive buy-and-hold strategy (Tsaih, Hsu, and Lai 1998). Chou et al. (1996) follow a similar approach for the Taiwanese market (Chou et al. 1996). Liu and Lee (1997) propose an Excel-based tool for technical analysis (Liu and Lee 1997). Other approaches combine neural networks with genetic algorithms (e.g. Baba and Inoue 2002). Kuo et al. (2001) show that they reach a higher prediction accuracy of stock development when including qualitative factors (e.g. political effect) in addition to quantitative data (Kuo, Chen, and Hwang 2001).

More interactive forms of decision support, where systems provide a laboratory-like environment for prospective investors to conduct standard as well as customized analyses have appeared. Dong et al. (2004) suggest a framework for a web-based DSS which implements a comprehensive approach to the investment task up to rebalancing an investor's portfolio according to his risk/return profile. They integrated On-Line Analytical Processing (OLAP) tools for customized multidimensional analyses (Dong et al. 2004). Another approach for interactive investor support provides the possibility of stepwise model generation such that investors can start with simple models from a toolbox and incrementally add more building blocks to arrive at more complex prediction models (Cho 2010). With a focus on more actionable support for private investors, Muntermann suggests a system design that estimates price effects of ad hoc notifications for public companies and sends out text messages to mobile phones including predicted effect size and time window (Muntermann 2009).

4.1.3 Purpose of this Paper

In summary, there is a broad range of investment decision support systems, ranging from providing better informational insight for the investor to very specific actionable investment support. To our knowledge, no investment decision support system uses

crowd votes from online stock community platforms to derive investment decisions even though research indicates that valuable insights can be expected. Therefore, we develop a system that is capable of supporting investors in their daily task of selecting promising stocks for their portfolio. Furthermore, the system is able to select stocks based on the crowd votes by an individual investment strategy and to transform those picks into a target portfolio. We provide first application results that demonstrate the advantages for investors to make use of such a DSS.

The remainder of this article is organized as follows: We present the decision support process and provide the system design that implements this process in section 4.2. The implementation of the design as a prototype can be found in section 4.3 and the evaluation of two test scenarios in section 4.4, while we summarize our contribution in section 4.5.

4.2 System Design

4.2.1 Methodology

In this paper we follow the methodology of Design Science Research as suggested by Hevner et al. (2004) which confounds an approach rooted in the engineering sciences. They provide several guidelines which we follow in the development of the proposed architecture (Hevner et al. 2004):

- **Design as an Artifact:** Design-science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
- **Problem Relevance:** The objective of design-science research is to develop technology-based solutions to important and relevant business problems.
- **Design Evaluation**: The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
- **Research Contributions:** Effective design-science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
- **Research Rigor:** Design-science research relies upon the application of rigorous methods in both the construction and evaluation of the design artifact.
- **Design as a Search Process:** The search for an effective artifact requires utilizing available means to reach desired ends while satisfying laws in the problem environment.
- **Communication of Research:** Design-science research must be presented effectively both to technology-oriented as well as management-oriented audiences.

The introduction and the theoretical foundation in section 4.1 show the relevance of the topic in the context of research. The relevance of the business problem is obvious,

as outperforming the market is both relevant to the whole financial industry and valuable given the profit prospects. Our artifact, the decision support system, addresses this problem and provides a solution (section 4.2) which is the result of the rigor search based on theory for a solution to the identified problem of how to make use of user-generated content for investment decision support. We evaluate our design and show its functionality in sections 4.3 and 4.4. In our concluding remark (section 4.5) we summarize our research contribution. All in all, this paper communicates our research to foster discussion as well as further research and applications in this area.

4.2.2 Crowd Stock Voting Data Structure

We used a data set of recommendations for a certain ISIN (International Securities Identification Number) on a certain day (called "votes") from an online stock community site. The main components of such a vote are: Stock Identification (i.e. ISIN), Recommendation (Buy/Sell), Start date (publication date of vote), End date, Start Price (price of stock at time of vote publication), Target Price (forecasted price), End Price (actual price at close of vote). The time from start date to end date is the runtime of a vote and it is called "open" on any date in between. Within this time frame, the target price is expected to be met. If the member who creates a vote does not specify an end date, the vote is automatically closed by the platform after 180 days. In addition, the member can close a vote at any time before the end date is reached. Recommendations can either be *buy* or *sell*, the target price accordingly is above or below the price at the publication of the vote (= start price). Figure 4-1 shows an example of such a user vote as it appears on the site.

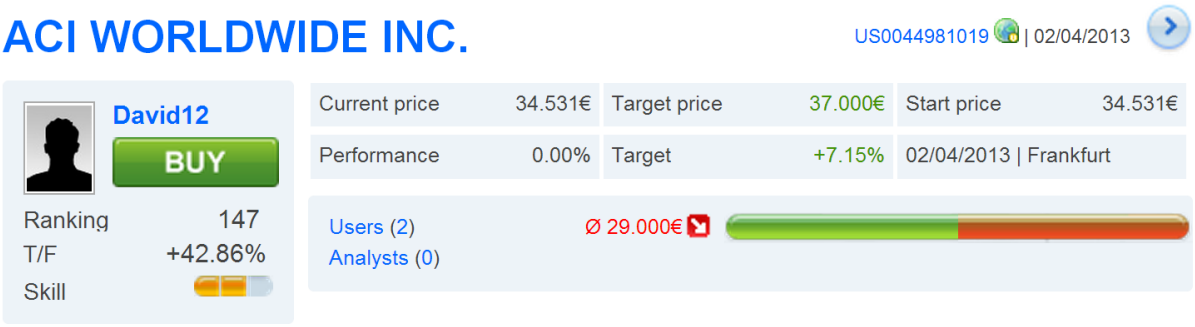

Figure 4-1. Example VIC vote

4.2.3 System Design and Decision Support Process

The system can support the investor in three different aspects: First, by creating a ranked list, the investor gets advice about the most preferable securities on a specific date. This information alone is already useful when selecting securities for a portfolio. In addition, the system supports the implementation and simulation of strategies

based on the computed ranking so that investors can test and explore different approaches to identify promising investment strategies. Once a suitable strategy has been identified, the system can be used to automatically follow a specified strategy day by day and create orders to modify a portfolio. This functionality helps to stick to a rational strategy and prevent investor's emotions to interfere and bias a well-defined strategy (cf. Lucey and Dowling 2005).

The system's task is to transform crowd votes into actionable share ratings for a given day. We split this task in two phases: 1) Rating Computation: Translate crowd votes into one or several performance estimate metrics per ISIN and 2) Investment Phase: Building a portfolio based on a list of shares ranked by such a metric, from which we select shares to create a portfolio. This approach follows the pattern of (Hill and Ready-Campbell 2011). Figure 4-2 gives an overview of the system design. The system executes both rating computation and investment phase for each day in the focus period.

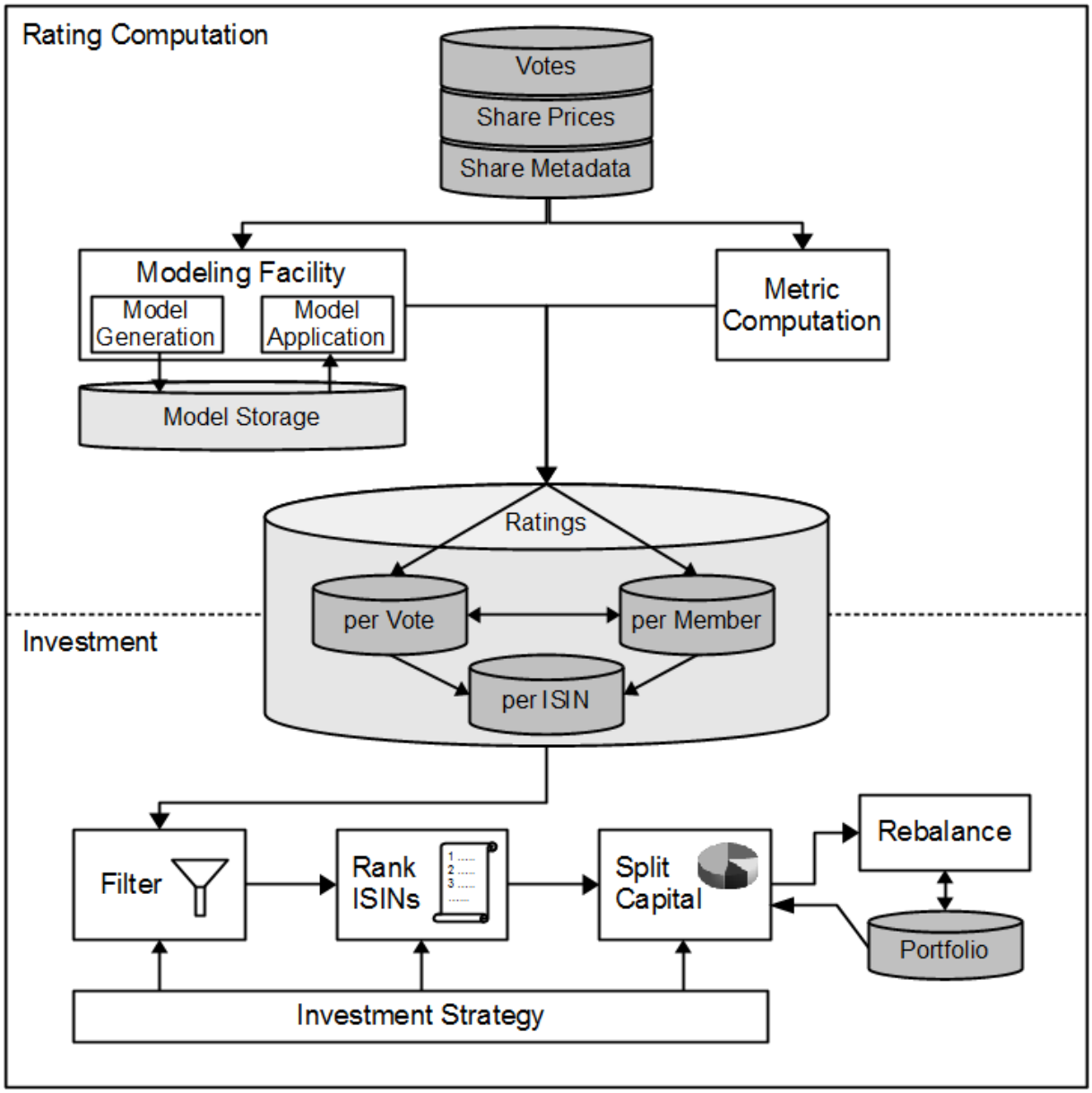

Figure 4-2. Overview of System Design

4.2.3.1 Rating Computation

The rating computation builds upon three data sources: the crowd votes, share prices for the period in focus and additional share metadata such as industry or geographic information. The system accesses these data either from a local database or online (e.g. via calls to APIs[10] of data providers). The term *rating,* as we use it, refers to the set of all computed metrics for a certain ISIN on a certain day. Which components of a rating are taken into account for portfolio creation depends on the investment strategy. The available metrics form a library of metrics as a basis for investment strategies.

[10] API = Application Programming Interface

Our design offers two approaches to arrive at a performance rating from the crowd votes. Simple metrics are computed by the system itself, without the need of external statistical software. By that, we avoid the overhead and effort of involving complex software packages when the metrics only involve simple average or sum calculations. However, in addition, we attach a modeling facility that supports the application of machine learning techniques on any data from the input sources. Hence the user of our system can use advanced modeling techniques to create complex models to support the investment decision (cf. Dong et al. 2004; Cho 2010; Groth and Muntermann 2009). When the user wants to create a model, s/he manually conducts a classical train-validate-test cycle on the data accessible by the system to arrive at a model suitable to her/his needs. After creation, a separate model store holds the models created by the user to apply them on new data and use its output as a metric in the rating computation process.

While metrics and models can be computed on any of the levels – vote, crowd member or ISIN level – we ultimately want to arrive at ratings on ISIN level as those are the building blocks for our portfolio. Hence, we need to transform all vote and member metrics into ISIN ratings. To give an example, a member ranking model delivers a metric that indicates the relative quality of a crowd member. Because we need to transform that ranking into an ISIN rating, we weight the votes of each member by the member's rank when aggregating the votes into ISIN ratings such that a vote of a higher ranked member has more weight than a lower ranked one. By doing so, we can combine metrics on different levels into a common metric on ISIN level.

The question of which metrics to combine, and how, depends on the application scenario and is hence subject to the user's experience or trial and error exploration. The system supports this task by making it possible to simulate investment strategies on any metric and thus enables a performance comparison based on a simulated portfolio.

4.2.3.2 Investment Phase

The second phase of the decision support process is the actual investment phase. Despite its purpose to execute a defined investment strategy to manage a portfolio, it also serves to simulate strategies and explore different approaches based on the ratings computed in the previous phase. The simulation approach is similar to that used by (Ruiz et al. 2012).

During the investment phase, the system transforms the stored ratings from the previous phase into a target portfolio. A portfolio in the system consists of a certain cash amount and a number of portfolio positions representing the shares held in the portfo-

lio. We call the set of parameters defining this transformation an *investment strategy*[11]. It defines which ratings have to be taken into account, how to rank them and how to split the available capital among the ranked ISINs. To modify a portfolio, the system creates and executes buy and sell orders.

During the filter phase, the system selects a subset of the available ISIN ratings as defined by the investment strategy. This filtering involves a diverse set of criteria, e.g. geographical scope, index membership, industry sector or conditions like “at least three crowd votes per rating”. Subsequently, the selected set is ordered by a chosen metric which is a component of the rating. If the user requires a ranking by more than one metric, we recommend to add a new metric which combines the other metrics in a defined way so that the decision criteria is transparent and documented for later reference.

After the ranking of ISINs, the available capital needs to be split as defined by the strategy. Example split strategies are: “Split evenly over top 10 ranked ISINs” or “Pick best ranked ISIN of three different industries.” The split affects the exposure to any individual ISIN and has to make a good trade-off between potential (focusing on less, but higher ranked ISINs) and risk reduction by diversification (spread capital over many ISINs to reduce correlation between shares). After computing the split, we arrive at a target portfolio composition. The system compares this target portfolio composition to the current portfolio and creates buy or sell orders in order to match the current portfolio to the target composition.

4.2.3.3 Implementation

We created an artifact to evaluate the functionality of the proposed decision support system design as a prototype in PHP. It has a console interface to enable batch mode operation in a real world scenario, but is also prepared to grant access via a web interface. It connects to a MySQL database that stores the crowd vote data, the stock quotes, the computed ISIN ratings and the portfolio data.

There are two modules for each of the phases “Rating Computation” and “Investment” which may run either together or independent. The rating computation module expects the time period for which to compute ratings as a parameter. The investment module demands a time period, a portfolio id and the ranking metric as parameters. Having two independent modules, the user can experiment with new metrics or modeling approaches while another user can run investment simulations. The modules are

[11] cf. Chapter 5

coupled via the common rating database. In the current state, the parallel usage of the modules is restricted to different time frames but we can easily solve this issue by duplicating the database, i.e. introducing staging areas.

4.2.3.4 Modeling Facility

For modeling, we provide an interface to the R software package[12]. We chose this software because it is easy to run in batch mode, has an open architecture including an extensive library for analytical methods and it is available without license cost. For model storage, we use the serializing features of R and store the models in separate files within the host's file system. Upon execution during the rating computation phase, the system takes care to set model parameters as defined by the user, provide the appropriate input data and collects the model results to write it back to the database.

4.2.3.5 Share Price Data

A financial website provides the quotes from the Frankfurt Stock Exchange for our system. These data includes daily prices (opening, closing, highest, lowest price and transaction volume) and the system uses them as follows: When an order is executed in a simulation run, the price of the share which is bought or sold corresponds to the next trading day's opening quote. This delay provides for the necessary time to collect all VIC votes on a day and then create orders for the next trading day (cf. Hill and Ready-Campbell 2011, 80). A daily system run cycle is not necessarily a restriction to the design, it can also be done weekly or monthly or several times during a day. The length of a reasonable interval depends on the frequency of changes in the crowd vote data set.

4.3 Prototype

In this section, we show implementation details of our prototype system to provide an application example of the proposed design. We show how to setup a simple investment support task, defining the metric and the investment strategy. Subsequently, we run a simple and a more complex test case on a test period of two years and present the results.

[12] http://www.r-project.org/

4.3.1 Definition of Performance Metric

In this section, we introduce the metric used to identify promising securities (ISINs) based on the wisdom of crowd concept. As mentioned, averaging the judgments of a crowd leads to superior estimates due to bracketing (Soll and Larrick 2009, 782). In terms of investment decisions, averaging several user votes whose target prices over- and underestimate the real target price might lead to a target price which is quite accurate. The metric introduced here, uses this effect based on the set of VIC votes for a certain ISIN on a certain day. It is a rather simple metric, so we are able to compute it without the use of the modeling facility.

To quantify the forecast of a vote, we compute the difference in price from the current quote at any given day t to the target price *PRICEGOAL* at the target date of the vote i for share *ISIN* with:

$$\Delta p_i(t) = PRICEGOAL_i - PRICE(ISIN, t)$$

If the vote is 100% correct, the price of the share will increase by Δp_i until the target date of the vote. When it comes to performance, another factor which has to be taken into account is the length of the forecast period. As mentioned, votes can have different runtimes and at any day there will be open votes with a higher or lower amount of remaining days. Logically, we prefer a 2% return in one week over 2% in one month. Earlier returns (which can be reinvested) should be valued higher over later returns of equal amount. We solve that by dividing the estimated price gap of each vote by the remaining runtime in days to arrive at a daily average expected performance:

$$E_i(t) = \frac{\Delta p_i(t)}{\Delta r_i(t)}$$

Δr_i simply is the remaining runtime in days at time t of vote i which ends at ENDDATE:

$$\Delta r_i(t) = ENDDATE - t$$

In addition, dividing by the remaining runtime days supports two other purposes: (1) as uncertainty of predictions increases the further they reach into future, the vote with the shorter and hence more certain time horizons are preferred, given equal absolute prospected returns, and (2) with regard to the specific data set, there are many votes that had no end date specified and hence were closed automatically by the platform after a timeout of 180 days. Hence, every vote that *does* have a specified end date has a runtime that very likely is less than 180 days. So given equal absolute price gaps, votes with specified end date (and hence a shorter runtime than the maximum runtime) are preferred over those open for a longer period. The assumption for doing

so is that a more specific vote is likely based on a deeper analysis or superior market insight.

To arrive at a specific investment decision, we roll up those forecasted average daily price changes of individual votes into a joint judgment for an ISIN. For example, if we have two buy and one sell recommendation for the same ISIN, the system combines the individual estimations into a "crowd" judgment by aggregating the individual price estimates, both positive and negative. Now to roll up all the individual estimates per ISIN, we take the mean of the daily expected returns derived for any ISIN from *n* votes:

$$E_{Perf}(t) = \frac{\sum E_i(t)}{n}$$

Table 4-1 shows a numerical example of the metric computation on a few sample records as it would turn out on February 5, 2013. The metric is pre-computed and stored in the database for any day and any ISIN appearing in open votes on that day.

Table 4-1. Calculation Example for the Rating Metric as evaluated on Feb 5, 2013

Vote	ISIN	End date	Δr (days)	Δp (EUR)	E_i	E_{Perf} (ISIN)
1	DE1111111111	10.02.2013	5	4.30	0.86	0.30
2	DE1111111111	05.06.2013	120	-6.00	-0.05	0.30
3	DE1111111111	01.03.2013	24	2.10	0.09	0.30
4	DE2222222222	06.03.2013	29	0.90	0.03	0.04
5	DE2222222222	24.05.2013	108	5.00	0.05	0.04
6	DE3333333333	18.02.2013	13	-3.50	-0.27	-0.07
7	DE3333333333	29.07.2013	174	0.70	0.00	-0.07
8	DE3333333333	30.04.2013	84	4.00	0.05	-0.07

4.3.2 Filter

We use the filter step to keep only ratings for shares listed in the German Prime Standard Indices, i.e. DAX, MDAX, SDAX, TecDAX – all other shares, including international and penny stocks were excluded from the set. We did that for two reasons: First, this ensures enough liquidity to successfully trade the identified stocks and second, to prevent the possibility of price manipulations by extensive pushing of certain (especially low-priced and low-volume) stocks.

In addition, we filter for ratings with a minimum number of votes (either sell or buy). This is to prevent single extreme votes from distorting the ranking and possibly keep sticking to the top of the list. For the scenario at hand, we require a minimum of three

votes for a rating – a deeper investigation on the optimal number of votes might yield further interesting insights.

4.3.3 Rank, Split and Rebalance

The system ranks the ISINs by ordering the remaining ISINs by E_{Perf} in descending order. This list is already an intermediate result which supports investors in their search of valuable shares using crowd wisdom. Within the system, the next step is to select ISINs from the list and split the available capital to create a target portfolio composition. For our test runs, we apply a simple and a complex split and rebalancing approach. Please refer to section 4.4.1 for details.

4.4 System Evaluation

4.4.1 Test Scenarios

We created two test cases which differ in the way the portfolio is created based on the list of ranked ISINs. Up until the creation of the ranked ISIN list, both test runs are equal.

The first strategy (Test 1) is rather simple, but rational: we invest all cash into the best-ranked ISIN. This is a risky, but also a promising approach. If the forecast is accurate, we maximize our profits by investing in the share with the highest return. We conduct a daily rebalancing of the portfolio: the system checks if the current ISIN in the portfolio is still the number one recommendation of the crowd and if not, it creates and executes a sell order for all of the shares in the portfolio and a buy order for the now best-ranked ISIN. If the portfolio is still empty, the system creates a buy order and invests the total cash position in the best-ranked ISIN. If the best-ranked ISIN is already in the portfolio from previous periods, no trade is initiated.

In a real-world application, investors would strive for a more diversified portfolio to reduce risk by combining shares which are unlikely to correlate in their development. In order to provide such a more sophisticated example of system usage, we demonstrate a second test scenario (Test 2) applying portfolio optimization (PO) on the crowd's Top10 selection of shares. Portfolio optimization, as introduced by Markowitz (Markowitz 1952), determines efficient portfolios for a given set of shares. Efficient portfolios either have a maximum return for a given level of risk (measured by standard deviation of historic returns), or a minimum risk for a given level of return. Using quadratic programming, portfolio optimization determines weights for each of the shares in the selection set to arrive at a well-diversified portfolio. More details on portfolio optimization can be found in (Markowitz 1952; Markowitz 1991; Andrecut

2013). For our purposes, we use an off-the-shelf implementation of the optimization algorithm in R provided by the *tseries* package (Trapletti and Hornik 2013). This also demonstrates the system's ability to include more sophisticated modeling facilities by interfacing the R universe with its extensive library of modeling and optimization methods.

To combine the PO with the crowd vote rating, we modified the investment strategy as follows: Instead of taking only the top ranked ISIN from the list, we take the top 10 shares and apply the portfolio optimization on those to determine the optimal weights for a target portfolio (which can also be 0, i.e. the algorithm may exclude shares). The portfolio optimization algorithm uses historic data of one year to determine the shares' volatility. To create a more common investment scenario, we also increased the rebalancing interval – i.e. the amount of days to pass before the portfolio structure is re-aligned to a given target composition – from one day to 20 (work) days, leading to a monthly re-arrangement of the portfolio structure. Less frequent rebalancing enables the portfolio positions to develop and gain the benefits from a correct investment decision. As less rebalancing means less trades, transaction costs are reduced as well. In addition, we set a minimum position size of 500 EUR (which was not needed in the first test scenario because there was only one position to invest) to avoid creating too small positions which would not be practical due to handling effort and transaction cost efficiency.

This scenario illustrates how the system is able to implement a more complex strategy containing three parameters which provide flexibility to find a suitable investment approach to users: the count of top shares of the ranked list that should be fed into the optimization run, the rebalancing interval in days, and the minimum position size per share.

4.4.2 Data

The vote data set used for our test run contains 15,727 crowd votes which were created between January 2009 and December 2010 from 1,353 distinct members considering 141 different shares. Table 4-2 summarizes some descriptive statistics.

Table 4-2. Descriptive Statistics of VOTE data set for test runs

Votes	absolute	in %
Average votes per day	21.6	
Number of distinct ISINs	141	
Number of members	1353	
Total number of votes	15,727	100%
By market segment (votes)		
DAX	5,647	36%
MDAX	4,589	29%
SDAX	1,815	12%
TecDAX	3,676	23%
By recommendation type (votes)		
Buy	9,169	58%
Hold	1,077	7%
Sell	2,693	17%
Strong Buy	1,968	13%
Strong Sell	820	5%
By year (votes)		
2009	5,976	38%
2010	9,751	62%

First, we let the system compute the metric ratings for the whole test period. After that, we conduct a full cycle of the investment phase starting from an empty portfolio with 100,000 Euro cash. The computation of the ISIN ratings for one day takes approximately 3-5 minutes on the test server (Intel i5-2520M, 2.5 GHz, 4 GB RAM) which is mostly spent on time consuming database queries. The investment simulation phase for Test 1 needs less than a second for a simulated day (i.e. build ranked list, pick first ISIN, build and execute order on simulated portfolio). For Test 2, the R run for the portfolio optimization takes a considerable amount of time (approximately 2-3 minutes including loading historic data from the database to compute the covariance matrix of share returns), but as we consider longer intervals this also leads to fewer optimization runs and thus to a slightly longer overall runtime than for Test 1.

For transaction cost (TC) calculation, we used the fee structure of a large German broker (comdirect Bank AG 2013). The per-trade cost is 4.90 EUR + 0.25%*Transaction Volume. The maximum cost charged for a transaction is 59.90 EUR (being effective at a transaction volume of and above 22,000 EUR which is the case for every of our simulated transactions). For a more transparent presentation, the transaction costs are not deducted from the cash account of the portfolio (and would hence reduce the subsequent cash amount available for reinvestment), but are rather modeled as being externally funded and thus only affect the final return.

4.4.3 Results and Discussion

In this section, we first describe the results of the rating computation which is equal for both test cases and produces the ranked ISIN list. Then we show and discuss the results of the two test scenarios. We then provide a comparison with two selected public funds and a common risk evaluation of all observed approaches.

4.4.3.1 Rating Computation

For the focal period, an ISIN rating bases on 15.71 votes on average. The top ratings, which ultimately determined the shares selected for the portfolio in Test 1, are based on 10.30 votes on average. This shows that not necessarily the number of votes determines the winning position. The average number of open votes for a day was 123.24 (Figure 4-3 shows the development over time) which means, on average, 123 ISINs were evaluated for a day. In a by-year comparison, this figure is rather low in 2009 with 110.53 and goes up to 136.05 in 2010. This shows that the analyst coverage increased due to increasing activity on the platform indicated by the growing number of user votes (cf. Table 4-2).

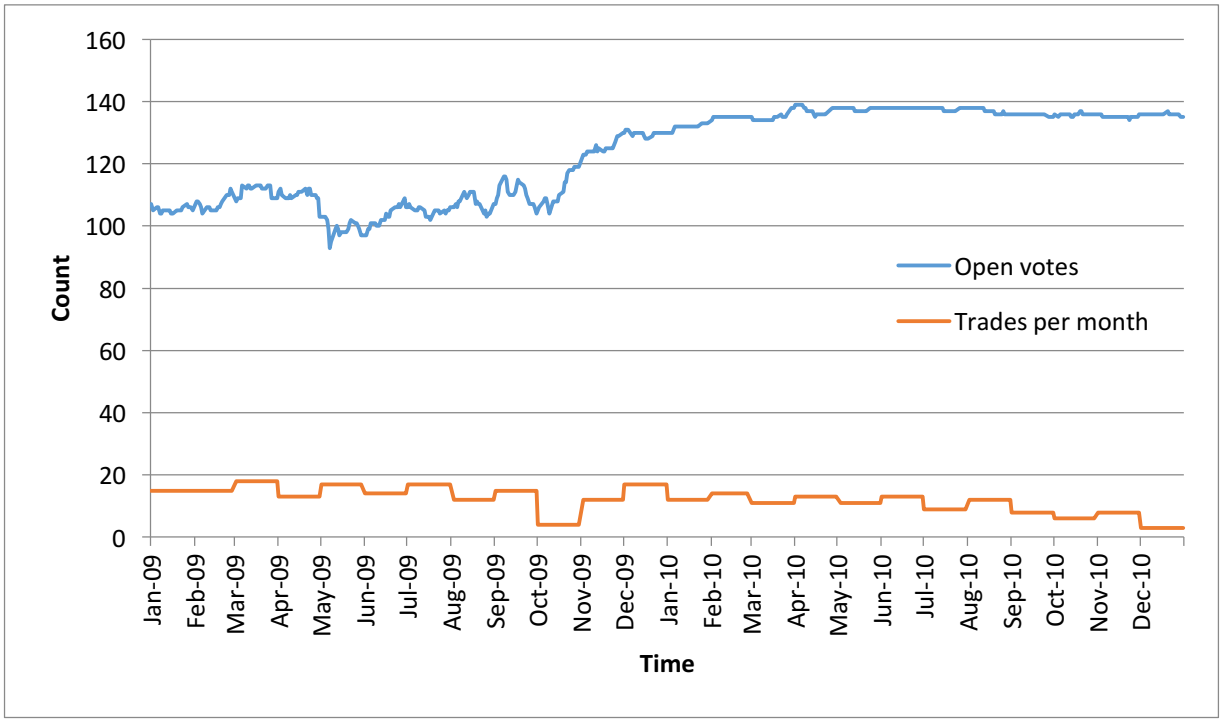

Figure 4-3. Development of the number of open votes and monthly trade volume

4.4.3.2 Results of Test 1

Figure 4-4 illustrates the investment simulation results for Test 1. The graph shows a standardized performance comparison of the portfolio performance, the portfolio performance after deduction of transaction costs and the performance of the DAX German market index (buy-and-hold). The lines indicate how an initial investment of

100,000 EUR develops over time according to the respective performance. Additionally, the chart shows the accumulated transaction cost which decrease the profits from the investment as can be seen by the spread between the two portfolio lines.

Table 4-3 provides an overall performance comparison which shows that the system outperforms a DAX index buy-and-hold strategy to a large extent. Even after considering the transaction cost, the performance is still roughly twice as large as the benchmark. In a daily comparison, on 275 of 520 days (53%) the crowd vote outperforms the DAX index benchmark. Interestingly, only 75 ISINs of the total of 141 (53%) appeared in first ranked positions at least once; the rest never made it to the top position.

Table 4-3. Results of Test 1: Comparison of Benchmark and Portfolio Performance incl. Transaction Costs

Instrument	Start January 2009 (EUR)	End December 2010 (EUR)	Absolute Return (EUR)	Return in %
DAX Benchmark	100,000	140,552	40,552	40.6%
Portfolio	100,000	223,164	123,164	123.2%
Portfolio – TC	100,000	188,484	88,484	88.5%

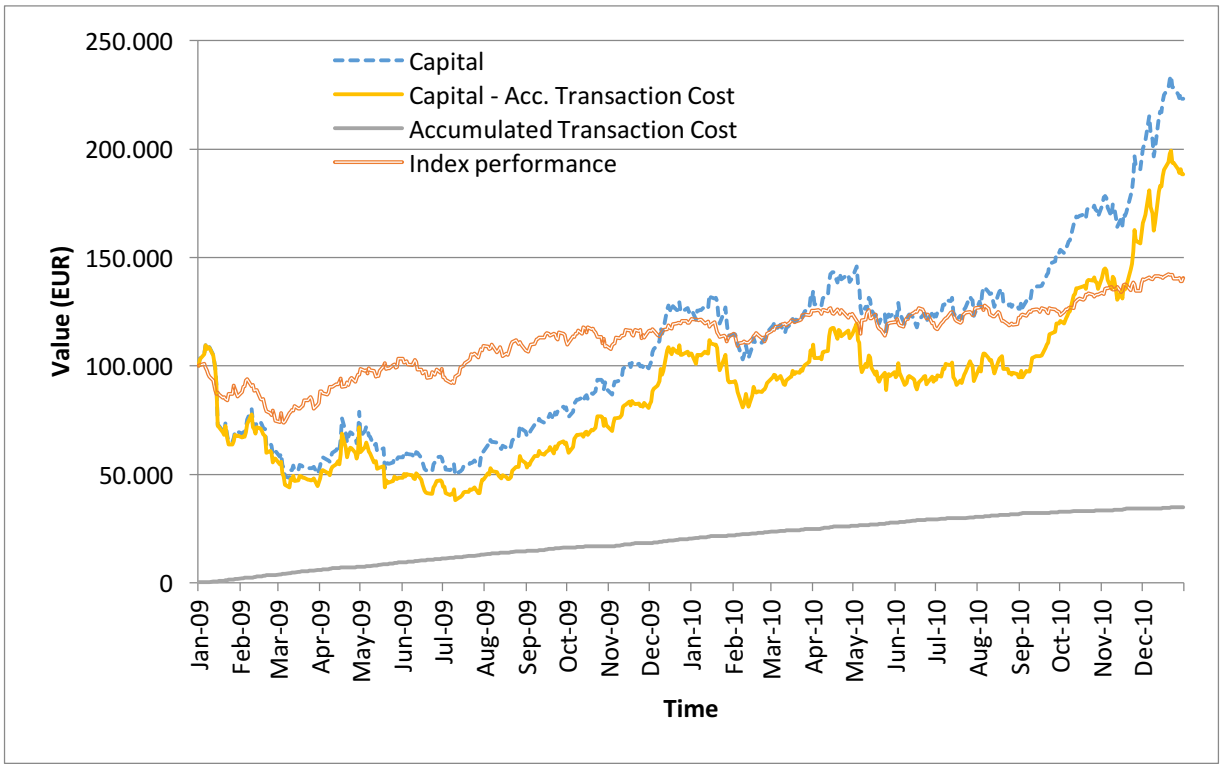

Figure 4-4. Results of Test 1 from January 2009 to December 2010

However, a closer look at Figure 4-4 reveals that the performance comparison can roughly be split in two phases: While the performance of the crowd votes is clearly underperforming the DAX index in 2009, the relation reverses in 2010 when the port-

folio starts to outperform the benchmark index. A possible explanation of why the performance increases in 2010 might be the increasing activity on the platform leading to a higher stability in crowd votes. As stated, Figure 4-3 indicates that the number of open votes for each day stabilizes on a high level in 2010. At the same time, the number of monthly trades triggered by the system enters a downward trend, also indicating a more stable crowd judgment leading to less fluctuation in the portfolio, even when checking daily for the single highest performing stock. These observations are only indicative and would need further analyses which are beyond the scope of this paper.

4.4.3.3 Results of Test 2

In Test 2 we apply a more sophisticated portfolio optimization (PO) on the crowd's Top10 selection of shares. This is a more realistic setup and depicts a scenario that might also be found in business practice.

Figure 4-5 shows that the diversified PO portfolio still outperforms the market index and reaches a performance of 70 points (without TC) or 59.2 points (including TC), respectively, above the benchmark (see Table 4-4.). In comparison to Test 1, the overall performance of the portfolio is slightly lower, yielding only a return of 110% compared to 123% in Test 1. Interestingly, after deduction of transaction cost, Test 2 portfolio still shows a return of 99.8% while the performance of the Test 1 portfolio drops down to 88.5%. This is caused by the fact that in Test 2 the portfolio is rebalanced on a monthly base compared to a daily rebalancing in Test 1 and thus the transaction costs are lower in Test 2.

The Test 2 portfolio outperformed the index benchmark on 265 of 522 days or 51%, respectively. The mean amount of positions in the portfolio is 4.77. This shows that the portfolio optimization algorithm is quite selective and reduces the number of possible positions on average by more than half (from 10 possible values). This includes the effect of a minimum position size of 500 EUR which prevents very low weights from the optimization run to result in a portfolio position.

Table 4-4. Results of Test 2: Comparison of Benchmark and Portfolio Performance incl. Transaction Costs

Instrument	Start January 2009 (EUR)	End December 2010 (EUR)	Absolute Return (EUR)	Return in %
DAX Benchmark	100,000	140,552	40,552	40.6%
Portfolio	100,000	210,600	110,600	110.6%
Portfolio – TC	100,000	199,824	99,824	99.8%

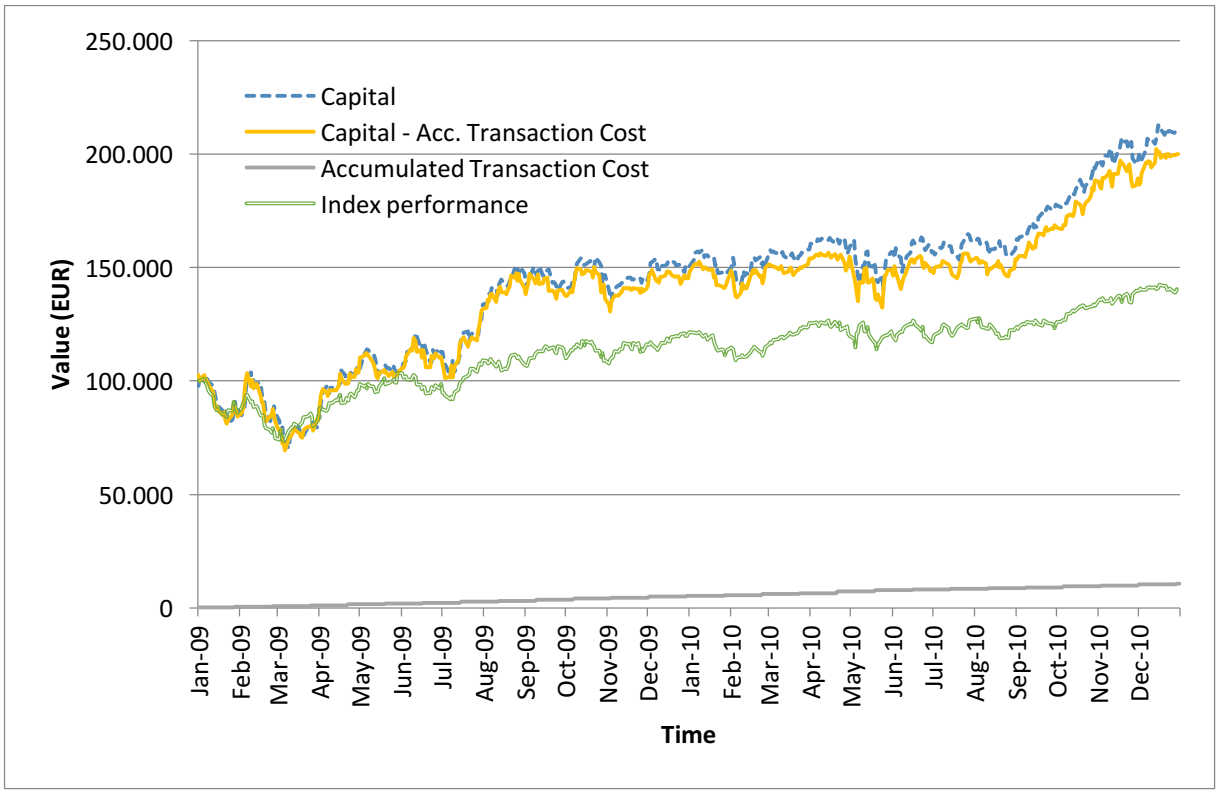

Figure 4-5. Results of Test 2 from January 2009 to December 2010

4.4.4 Performance Comparison with Public Funds

We compare the system results with the performance of two selected public funds over the same period. Such funds provide another benchmark for the performance of the two test runs and gives a reference to evaluate the external validity of the system results. As public funds are managed by professional institutions with a defined investment strategy and a specific analytic approach, they can serve as a state-of-the-art benchmark for our DSS which tries to solve the same task.

To have a comparable benchmark, we have to choose funds that meet some requirements: the portfolio must pick shares from a similar investment universe (i.e. German Prime Standard), provide sufficient historic data for our test period and should be representative for their kind, i.e. have a high reputation or be important in terms of size. Based on these prerequisites, we chose the following two funds for comparison:

- **Fund 1: DWS Deutschland (ISIN DE0008490962)**: One of the largest funds for German Standard shares (3.3 billion EUR in assets), Morningstar Silver Rating (Nöth 2013). The strategy of portfolio building is a mixture of top-down elements (e.g. sector allocation, share of mid/small caps) and bottom-up (strong emphasis on fundamental analysis using standard valuation models, management briefings and common stock metrics (P/E & P/B ratio, EBITDA), among others). The investment ratio depends on technical indicators, sentiment and

general assumptions about the market development (Nöth 2013). This fund is representative for a large, traditionally managed standard fund.

- **Fund 2: Allianz Thesaurus AT EUR (ISIN DE0008475013)**: This fund (153 million EUR in assets) is based on a momentum model that tries to identify trends to over- or underweight certain shares from the investment universe. The model also considers risk contribution of shares (Claus 2013). We chose this fund to compare our test results to a professional approach based on quantitative modeling.

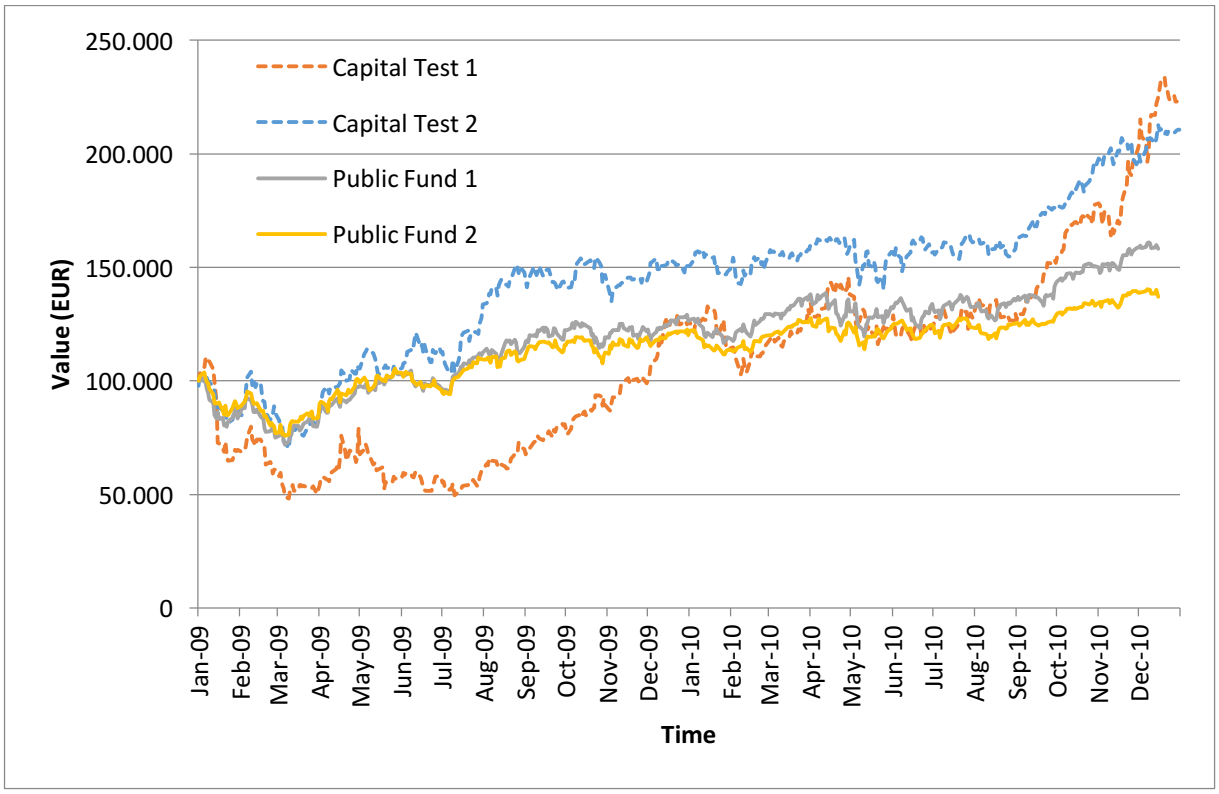

Figure 4-6. Comparison of test portfolios and public funds

Figure 4-6 shows that the two public funds have approximately the same performance at the beginning of our observation period, but start to diverge at the end of 2009, with Fund 1 performing superior until the end of the test period. The Test 1 portfolio clearly underperforms all alternative portfolios in 2009, but rises up to the top until the end of 2010. The optimized portfolio of Test 2 shows a better and steadier performance, climbing to the top early and staying there almost until the end of the period. However, the performance differences might come with different portfolio risks which we will hence examine in the following section.

4.4.5 Risk Evaluation

A common measure for the risk associated with the investment into a certain security is the volatility of its returns. Hence, to judge the risk of a security, we compute the

standard deviation of security prices (Hull 2007; cf. Sharpe 1992a). When two securities yield the same return, the one with the lower volatility had a more steady development and as such a lower risk. Usually, higher return potential comes with a higher risk. To measure the "price of risk" and make investments which differ both in potential and risk comparable to one another, Sharpe (Sharpe 1966) introduced the "Reward-to-Variability-Ratio" (Sharpe Ratio). This ratio S of an investment is defined as:

$$S = \frac{\overline{D}}{\sigma_D}$$

$\overline{D}$ is the average outperformance against a riskless investment over a time period divided by the standard deviation σ_D of these outperformance returns during the time period (Sharpe 1992a). The Sharpe Ratio indicates the reward for each unit of risk and makes different investments comparable. A higher Sharpe Ratio indicates a better return-to-risk relation and hence a better investment. In our work, we assume there is no risk-free alternative to stock investment (we force all cash to be invested into shares) in order to focus on examining the quality of the stock selection process, i.e. we use absolute returns for the Sharpe Ratio.

Table 4-5 summarizes the Sharpe Ratio and its components for all portfolios examined in this paper. We consider all portfolio performances before transaction costs because we do not have information on the transaction costs of the public funds and for replicating the DAX benchmark. Comparing the results of Test 1 and Test 2, we see that the mean daily return of Test 1 is higher, but this also holds for the standard deviation and hence – volatility or risk. The Sharpe Ratio of Test 1 Portfolio (0.0648) is lower than those of Test 2 (0.0791) which means that the PO was a valuable strategy, because it reduced risk more than it reduced returns. Looking at Figure 4-6, we see that the Test 2 portfolio avoids the low trough of the Test 1 portfolio in 2009, but also exhibits more sideward development later on while Test 1 portfolio increases rapidly in 2010. Figure 4-6 thus nicely illustrates the reduced volatility of the Test 2 portfolio.

The public fund 1 has a lower return than the system portfolios, but also a lower risk, leading to a Sharpe Ratio of 0.0604 which is above the benchmark, but still below the two system portfolios. Hence, the return loss is not completely compensated by additional reliability in comparison to the test portfolios of the system.

Public fund 2 shows a worse performance for the test period for both return and risk, even below the benchmark. It shows for the test period that an approach based on a quantitative model implementing a momentum strategy performs below the results of our test runs.

Both public funds have a lower risk for the test period, indicating the strict risk management of public funds compared to our simple test approaches. In addition, both funds tend to a high share of large caps (Nöth 2013; Claus 2013) which make them stick close to the DAX benchmark (which contains the 30 largest public companies of Germany). In contrast, our system's approach did not prefer large caps over small or mid caps but let alone the crowd's estimate determine preferences. If we focus however on absolute returns and/or on the Reward-to-Variability-Ratio, our Test scenarios, especially Test scenario 2, clearly outperform all other benchmarks.

Table 4-5. Mean and standard deviation of daily returns, Sharpe Ratio for all observed portfolios (Jan 2009 – Dec 2010)

Portfolio	Mean of daily return	Standard Deviation of daily return	Sharpe Ratio
DAX Benchmark	0.0009	0.0150	0.0581
Portfolio Test 1	0.0021	0.0321	0.0648
Portfolio Test 2	0.0016	0.0208	0.0791
Public Fund 1	0.0011	0.0174	0.0604
Public Fund 2	0.0007	0.0155	0.0476

4.5 Conclusion

Research shows many approaches for investment decision support systems but none of them enables users to take advantage of the wisdom of crowd even though it has been shown to be advantageous to investment decisions (Hill and Ready-Campbell 2011; Avery, Chevalier, and Zeckhauser 2011) for the price of complex analytical effort. We address this gap and propose the presented system design; it provides flexible means to transform raw crowd vote data into actionable investment decisions, up to automatic portfolio maintenance. By building a prototype and running two test cases that clearly outperformed a market benchmark index over a period of two years, we showed the viability of our concept. Even with a rather simple strategy, we achieve a portfolio performance of 123%, outperforming the market benchmark by 83 points and still 48 points after transaction costs. A second test case illustrated how a sophisticated optimization approach can be included to consider risk and build a diversified portfolio that still outperforms the market benchmark by 70 or 60 points (after TC), respectively. A comparison with public funds shows that the results are reasonable given the institutional boundaries of such funds compared to the prototypical character of our test scenarios. Further steps have to be conducted before a real-world application of the system is deemed appropriate. Specifically, further testing in different market phases and an appropriate strategy formulation[13] as well as adaption to real-

[13] See Chapter 5.

world restrictions in portfolio building is necessary. Minimum position size and adjustable rebalance intervals are already first steps in fulfilling those requirements.

Both cases show that the proposed decision support system enables investors to make use of the wisdom of crowd and automatically test and apply strategies based on this source. The results of our system can either be used standalone or be integrated with other traditional measures from fundamental or chart analysis. It could serve as an additional signal when it comes to an investment decision with respect to a certain stock.

From a scientific perspective, the system provides a laboratory environment to further analyze the phenomenon of crowd wisdom in financial markets. The infrastructure provided by the system, enables us to further investigate and simulate promising models and investment strategies. One possibility to arrive at the best strategy would be to create numerous software agents that try to maximize their outcome by using the proposed DSS (cf. (Hinz and Spann 2010)). Such a multi-agent simulation would help to increase our understanding on the exploitation of the wisdom of the crowd in financial markets.

Future improvements of the proposed system could include the introduction of a target risk to guide the decisions proposed by the system and match them with an investor's risk profile. Additionally, literature suggests that the predictive power of crowds can be increased by selecting a group of experts based on historical performance within the crowd (Hill and Ready-Campbell 2011). This could be taken into account by creating a ranking metric that gives more weight to potentially more experienced crowd members. The modeling facility of our proposed system already allows incorporating the member experience in the ranking models. However, since this approach contradicts the wisdom of crowd concept which strives for maximum diversity (Surowiecki 2005), it is a different approach than used in the application example of this paper, but is supported by the flexibility of the system design and worth further exploration.

5 A Formal Model for Investment Strategies to Enable Automated Stock Portfolio Management

Title	A Formal Model for Investment Strategies to Enable Automated Stock Portfolio Management
Author(s)	Gottschlich, Jörg, Technische Universität Darmstadt, Germany Forst, Nikolas, Technische Universität Darmstadt, Germany Hinz, Oliver, Technische Universität Darmstadt, Germany
Published in	Proceedings of the International Conference on Information Systems (ICIS) 2014, December 14-17, Auckland, New Zealand VHB-Ranking A

Abstract

In this paper, we develop a formal model to specify a stock investment strategy. Based on an extensive review of investment literature, we identify determinants for portfolio performance – such as risk attitude, rebalancing interval or number of portfolio positions – and formalize them as model components. With this model, we aim to bridge the gap between pure decision support and algorithmic trading systems by enabling the implementation of investment approaches into an executable specification which forms the foundation of an automated portfolio management system. Such a system helps researchers and practitioners to specify, test, compare and execute investment approaches with strong automation support. To ensure the technical applicability of our model, we implement a prototype and use it to show the effectiveness of the model components on portfolio performance by running several investment scenarios.

Keywords: Investment Decision Support, Investment Strategy, Stock investment, Portfolio Management, Investment Strategy Model

5.1 Introduction

IT increasingly dominates international financial markets and stock exchanges. While stock trading *transactions* have taken place via electronic networks for quite some time, recent developments show an increasing automation of the *decision making* processes when it comes to stock investments. The advantages are obvious: machines are able to conduct large-scale analyses 24/7 and neither ask for commissions nor do they have hidden agendas. Once an investment strategy is defined, they conduct it rigorously, which might sometimes lead to undesired results such as the Flash Crash on May 6, 2010 (Patterson 2012). But compared to human investors, machines are immune against the influence of emotions which have shown to spoil the objective judgment of humans (Lucey and Dowling 2005).

However, the human investor is still needed to identify investment opportunities and define strategies to exploit those opportunities. It seems reasonable to combine the strengths of both actors: use the human creativity and experience to identify and define investment strategies and then let a machine execute this strategy automatically to avoid human emotions conflicting with rational strategic decisions. Not surprisingly, research suggests a range of system designs to support investment decisions (see the following section for an overview). However, we did not find a flexible and comprehensive framework to specify an abstract investment strategy in a form machines can understand and execute. Such a generic model serves researchers as well as practitioners as a conceptual foundation for portfolio management systems. For researchers, an implementation of the model helps to build a laboratory environment to analyze investment strategy patterns and test new approaches for stock investment analysis (e.g. using indicators from Twitter or News analyses). By allowing batch runs on historic stock data with different parameter settings, researchers can estimate parameter effects on portfolio performance by conducting sensitivity analysis. For practitioners, the model allows a precise specification of an investment idea including backtesting and provides strong support to automate portfolio management.

In this paper, we develop a formal model to specify a stock investment strategy. Based on an extensive review of investment literature, we identify determinants for portfolio performance and describe them formally to fit in our model. We build the model with the intention to enable the implementation of an automated portfolio management system, i.e. a system that is able to receive a formal strategic description and act upon it autonomously for a specified period of time (either a simulation on historic data or a real-time application with current stock market data). The challenge is to find a way of specification which is universal and flexible enough to cover the broad range of possible approaches for investment strategies, but yet formal enough to be executed

by a machine. To ensure a maximum flexibility, we integrate a generic stock ranking mechanism (cf. Gottschlich and Hinz 2014 and section below) based on user-defined metrics derived from stock analysis (e.g. momentum or volatility). Such a metric or a combination of several metrics is used as a score to bring the stocks of the investment universe in an order of preference. Hence, we are able to include a flexible way to express for any investment strategy what makes a stock more preferable over another form the investor's point of view. We ensure that our model can be implemented in software by implementing our model specification as a prototype system and use the results to show the effectiveness of the model components.

Previous approaches focus on specific aspects of investment decision making, e.g. security price analyses or news effects (e.g. Muntermann 2009), but an overarching framework to integrate those techniques in a flexible way and apply them for automated portfolio management is not yet available. Application scenarios for such a model are plentiful though: Investors can simulate and test strategic approaches to stock market investments based on a variety of indicators on historic data. Once they find a working strategy, they can enable automatic strategy execution for future transactions within the parameters specified by the investment strategy. Our approach targets at the gap between professional algo- or high-frequency trading of large institutions and manual stock selection of private investors. The main intention is to make the analytical capabilities (and to a less extent the speed advantages) of information systems accessible for stock market investment decisions in a flexible and agile way to provide a laboratory environment and decision support system for investors (cf. Gottschlich and Hinz 2014). This paper introduces the model as a "language" to express the reasoning and restrictions of an investment strategy, but does not aim at finding optimal parameter settings in terms of portfolio performance. In fact, as we designed the model to be generic for a large diversity of investment approaches, this would result in the search for an optimal investment approach to stock markets – though desirable, this exceeds the scope of this paper.

The remainder of the paper is organized as follows: the upcoming section provides an overview of previous works in the area of investment decision support. Subsequently, we introduce our methodology. The section "Model Development" introduces all model components based on an extensive body of investment literature and closes with a formal specification of our model. The extensive evaluation of the model functionality based on six scenarios follows in the "Model Evaluation" section. We conclude our work with the final section showing venues for further improvement of this approach.

5.2 Investment Decision Support

The majority of previous research on investment decision support strives to provide better insights to investors by improved information support. Commonly used are methods to model stock price development utilizing optimization or machine-learning approaches. Specifically, artificial neural networks show broad coverage in investment decision support literature, often in combination with other approaches. Tsaih et al. (1998) combine a neural network approach with a static rule base to predict the direction of daily price changes in the S&P 500 stock index futures which outperforms a passive buy-and-hold strategy (Tsaih, Hsu, and Lai 1998). Chou et al. (1996) follow a similar approach for the Taiwanese market (Chou et al. 1996). Liu and Lee (1997) propose an Excel-based tool for technical analysis (Liu and Lee 1997). Kuo et al. (2001) show that they reach a higher prediction accuracy of stock development when including qualitative factors (e.g. political effect) in addition to quantitative data (Kuo, Chen, and Hwang 2001).

More interactive forms of decision support have appeared, e.g. systems providing a laboratory-like environment for investors to conduct standard as well as customized analyses. Dong et al. (2004) suggest a framework for a web based DSS which implements a comprehensive approach including support for rebalancing an investor's portfolio according to his risk/return profile. They integrated On-Line Analytical Processing (OLAP) tools for customized multidimensional analyses (Dong et al. 2004). Another approach for interactive investor support provides the possibility of stepwise model generation such that investors can start with simple models from a toolbox and incrementally add more building blocks to arrive at more complex prediction models (Cho 2010).

The impact of news on stock prices can lead to distinct market reactions and hence support for their evaluation has been subject to several works. Mittermayer (2004) suggests a system which processes news to predict stock price movements taking a three step approach: extract relevant information from news by text processing techniques, assign them into three categories (good, bad neutral) and then turn those into trading recommendations (Mittermayer 2004). Schumaker and Chen (2009) use a similar approach of news analysis applying different linguistic textual representations to identify their value for investment decisions (Schumaker and Chen 2009). Muntermann (2009) focuses on more actionable support for private investors and suggests a system design that estimates price effects of ad hoc notifications for public companies and sends out text messages to mobile phones including predicted effect size and time window (Muntermann 2009). This helps private investors to decrease disadvantages in speed or awareness they usually suffer compared to institutional investors.

Other approaches try to exploit collective intelligence for stock investment decisions. Stock recommendations from stock voting platforms on the Internet have shown to be a valuable source for investment decisions (Hill and Ready-Campbell 2011; Avery, Chevalier, and Zeckhauser 2009) and can even be superior to the advice of institutional analysts (Nofer and Hinz 2014). Therefore, Gottschlich and Hinz (2014) suggest a decision support system which uses the wisdom-of-crowd concept to automatically select stocks from a German market and transform this selection into a target portfolio based on a formal investment strategy. While their approach is close to an automated portfolio management system, their specification of a formal investment strategy is still at an early stage and not extensively rooted in investment literature inducing a need for further development.

Surprisingly, there is little coverage of algorithmic trading system designs in scientific literature. This might be explained by the distinct value such designs have to their (commercial) developers, who have little incentive to spread this valuable knowledge. However, some recent works also shed some light onto this field (Kissel 2013; e.g. Narang 2009).

Based on these streams of literature, we are not aware of an integrated model to express investment strategies in a formal way while allowing for generic and hence flexible investment potential identification. In this paper, we want to address this gap by providing a formal model which an investor can use to specify an investment strategy in a formal way so it can be processed by an automated portfolio management system. We strive for a systematic approach of combining decision support of investors (i.e. providing insight or information) and stock investment execution (i.e. buying and selling selected stocks). Such a system derives a target portfolio layout by applying the formal investment strategy on specific stock data and provides simulation of an investment strategy on historic data or trading a strategy on current data. Our core question is: How can we formally specify investment strategies to enable automatic strategy execution? This model is supposed to be a major building block for the development of an automated portfolio management system.

5.3 Methodology

We conduct an extensive literature review of investment literature. From literature, we identify factors that drive portfolio performance which we formalize and integrate into our model. The goal is to create a universal model which bridges the gap between the domains of investment strategy formulation and the detailed instruction level a machine needs to process.

To ensure our model is really executable by a machine, we implement a prototype in the validation section and analyze the effectiveness of each model component using a scenario analysis. Before we begin to develop the model, we introduce two central terms needed for understanding throughout this paper.

5.3.1 Risk measure

The risk of a portfolio is the probability of missing an expected return. We can split the total risk of a portfolio into two components: systematic and unsystematic risk (Evans and Archer 1968; Y. Li et al. 2013). Unsystematic risk evolves from each of the individual positions in the portfolio and is related to the specific company (e.g. management errors). An investor can reduce this kind of risk by diversifying his portfolio (Y. Li et al. 2013; Statman 1987). In contrast, systematic risk affects the market as a whole (e.g. change in interest rates) and is not affected by portfolio diversion.

We model the risk associated with an investment in a certain security as the volatility of its past returns measured by the standard deviation. This is a common approach in literature (Sharpe 1992b; Markowitz 1991) and easy to understand: if we have two securities yielding the same return, the one with the lower volatility showed a more steady development and hence a more reliable return potential throughout the holding period.

5.3.2 Profit/Return measure

The profit or return of an investment is the percentage of value increase (or decrease, if negative) within a specific period. Formally, we measure profit R_t as:

$$R_t = \left(\frac{p_t}{p_0} - 1\right) * 100$$

with p_t = price of stock at end of period t, p_0= price of stock at beginning of period t. Intuitively, we measure return as the difference of buying and selling price, while disregarding distributed dividends.

5.4 Model Development

In this section we are going to introduce the components of our formal investment strategy model based on an extensive investment literature research to identify factors relevant for the performance of a portfolio (cf. Table 5-1, p. 91). These components serve as a parameter set which jointly describes an investment strategy in a formal way. One instance tuple of these parameters describes a specific strategy which an

appropriate software implementation can interpret and process. We integrate each of the identified components into our formal model which we specify subsequently. Figure 5-1 shows an overview of the investment process steps that implement such a strategy and the application of the model components in each step. The process starts with a computation of a ranking metric from stock data which is subsequently used to rank the stocks of the investment universe considered by the investor. Based on this order of preference, the system splits the available capital, determining the sizes of each portfolio position. Finally, the portfolio is rebalanced such that it reflects the new layout defined in the process.

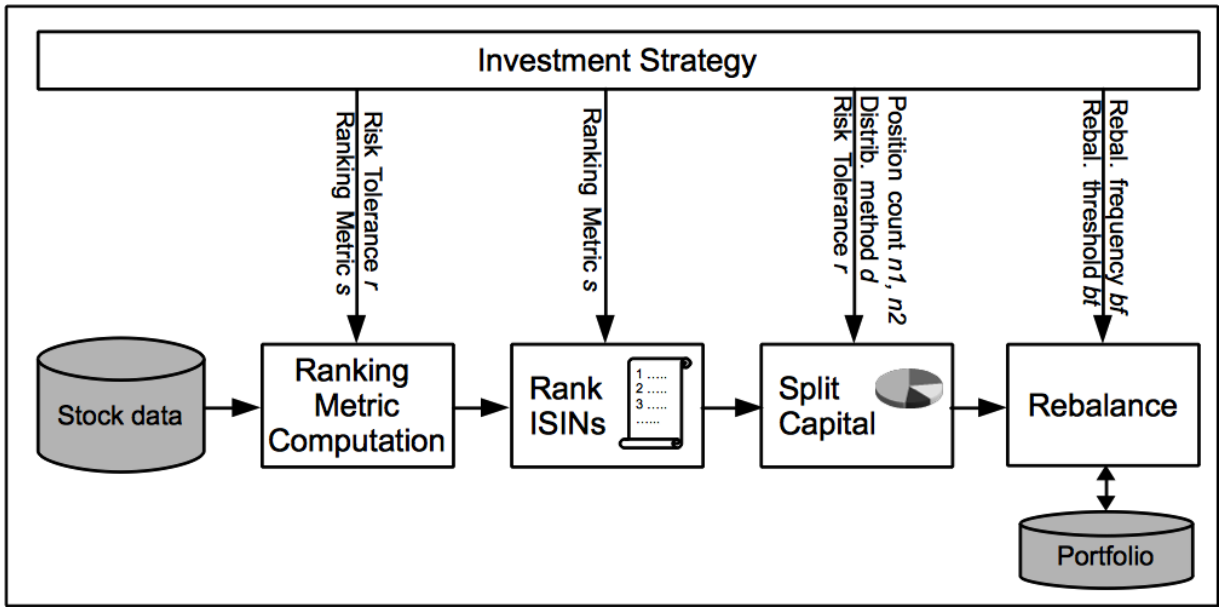

Figure 5-1. Overview of Investment Process and Influence of Model Components on Process Steps

5.4.1 Investor's Risk Tolerance

One of the fundamental parts of an investor's strategy is the definition of the target risk-return profile. The relationship between risk and return has been extensively discussed in literature (cf. Guo and Whitelaw 2006) and there is evidence of a trade-off between risk and return (Ghysels, Santa-Clara, and Valkanov 2005; Guo and Whitelaw 2006). Intuitively, it makes sense that investors demand a higher potential return to invest in a stock with a higher risk. Modern Portfolio Theory states that investors strive to create a portfolio that maximizes returns at a given risk level or to minimize risk at a given return target (Markowitz 1952). Hence, it is sufficient to define either a target risk or a target return.

We decide to model the risk-return profile as a maximum acceptable risk level and strive to optimize portfolio return. This is more intuitively to investors and more convenient when it comes to implementation because we can compute a good estimate of

a security's risk based on past development (compared to the difficulty of forecasting expected returns precisely). Of course, the risk computed from the security's historic development might not be an appropriate predictor for its future risk – but this is a general problem in investing which we cannot address ex ante with our model.

Technically, the risk level specified with the investment strategy by the investor serves as the target risk for the portfolio creation/rebalancing step. Based on this restriction, the goal is to maximize the prospected return. At the current stage, we model the risk tolerance as the amount of volatility (based on stock price history) an investor is willing to take when a portfolio is created. When using the Markowitz portfolio optimization (see section 5.4.5.2), we use the risk tolerance as target risk level which forms a restriction for the portfolio optimization algorithm. Doing so, the risk level of the target portfolio should likely be in the desired range of the investor. Formally, we denote the investor's risk tolerance with $\boldsymbol{r}$.

This modeling of risk as threshold to volatility is a quite technical and simple approach. More sophisticated methods for risk management have been proposed and enable investors to express the risk they take more precisely. Specifically, Value-at-Risk (VaR) is a common measure to express risk associated with an investment (see e.g. (Linsmeier and Pearson 2000) for an introduction). VaR is the amount of loss which will be exceeded only with a specified probability p within an observation period t. There are several common methods to determine VaR: The Historical, the Delta-Normal and the Monte Carlo Approach (for details see Linsmeier and Pearson 2000). They have in common, that they apply a distribution of profits/losses of an investment to determine an amount of loss that is only exceeded with a defined probability, usually 5% or less. For example, if we look at the historic distribution returns for an investment within the period t, the amount at the 5% quantile is what an investor would lose if the return would be as bad as it has only been in 5% of historic cases. The rationale is that under "normal circumstances", i.e. within the 95%, the loss will be less than the VaR and hence within that confidence, risk is under control. The definition of p demarks the border between "normal" and "abnormal" circumstances and depends on the investment universe and the attitude towards risk of the investor. For more details on how to use VaR for portfolio management, see e.g. (Krokhmal, Palmquist, and Uryasev 2002) or (Ogryczak and Sliwinski 2010).

To apply VaR for our model as an alternative to the target risk level $\boldsymbol{r}$, we need to exchange r with the parameters necessary to determine VaR: $\boldsymbol{p}$ which is the quantile to determine the VaR value from the profit/loss distribution and $\boldsymbol{t}$, the holding period. In our model, the holding period is connected to the rebalancing interval (see section

5.4.2) because in between rebalancing events, no changes in the portfolio will occur. This has to be taken into account when using the model for implementation.

Regarding the method of VaR calculation, we suggest using historical VaR if the necessary data is available as it should deliver highest accuracy for the price of computation effort – which is convenient for our intention to create a machine executable model for investment strategies. Alternatively, Delta-Normal VaR is a simplified method of computation which should be preferred in a portfolio management system if performance is an issue.

5.4.2 Rebalancing Interval or Frequency

Markowitz initially specified his model only for the case of one period to set up an optimal portfolio at the beginning (Elton and Gruber 1997), but disregards actions necessary to contain risk or secure intermediate returns over the course of time, such as checking and re-adjusting portfolio positions (Cohen and Pogue 1967). However, in reality, when portfolio positions shrink or grow over time due to their price development, frequent checking and rebalancing of portfolio positions is important to restore the initial asset allocation and meet the target profile of the portfolio regarding diversification and risk (cf. Buetow et al. 2002). Such adjustments lead to sell or buy orders for portfolio positions which cause transaction cost. Hence, an investor needs to maintain a balance between necessary rebalancing and cost for adjustment (Woodside-Oriakhi, Lucas, and Beasley 2013).

There are two basic approaches to portfolio rebalancing: *Calendar/frequency* rebalancing and *range* rebalancing (cf. Buetow et al. 2002; Plaxco and Arnott 2002). Using *calendar rebalancing*, an investor checks frequently, e.g. every month, quarter or year, the positions of his portfolio and adjusts them according to the target layout. This is an easy approach which does not depend on any external triggers, but is also passive towards dramatic changes in market environment as long as they happen in between rebalancing intervals.

Range rebalancing, in contrast, defines thresholds which specify the tolerance of an investor towards changes in position size. For example, a tolerance of 5% allows position sizes to deviate up to 5% from the initial asset allocation (up or down) before a rebalancing of the portfolio is triggered (Buetow et al. 2002). This allows investors to react immediately on possibly serious changes in portfolio positions and thus limit losses close to the tolerance interval. However, in reality, this approach needs support by suitable automatic portfolio surveillance to enable timely reaction.

Plaxco and Arnott (2002) show how important rebalancing is to maintain a defined risk profile: starting from a portfolio split of 50% in equities and 50% in bonds in 1950 and following a *drifting mix* strategy (i.e. re-investing any dividends in equities and interest in bonds), the portfolio would end up at a split of 98% in equities and 2% in bonds over the course of 50 years – resulting in an obviously different risk profile (Plaxco and Arnott 2002). Buetow et al. (2002) tested a combined approach of *calendar* and *range rebalancing*. They defined a threshold for position size deviation and checked this threshold frequently after a defined time period. They found that especially when markets are turbulent, frequent rebalancing is profitable (Buetow et al. 2002). In addition, in most cases range-based rebalancing had positive effects on portfolio performance over the last five decades (cf. Plaxco and Arnott 2002).

Thus, rebalancing is important to maintain the intended risk profile of an investor over time. We introduce both *calendar* and *range rebalancing* into our model and denote the rebalancing frequency with ***bf*** and the threshold for deviations of portfolio size in % with ***bt***.

5.4.3 Number of Portfolio Positions

The number of positions in a portfolio is another determinant of portfolio diversification. Statman (1987) found that a portfolio's risk – measured as the standard deviation of returns – drops with every additional random stock that is added (Statman 1987). Elton and Gruber (1977) found that that 51% of a portfolio's standard deviation can be eliminated by increasing the number of positions from 1 to 10. Additional 10 positions only eliminate another 5%. Figure 5-2 shows the decline of standard deviation (risk) with an increasing amount of stocks in the portfolio based on a correlation between stocks of 0.08 (as measured by Campbell et al. (2001)). It confirms that the major part of standard deviation reduction can be achieved with 10-20 stocks. Statman (1987) recommends 30-40 different positions.

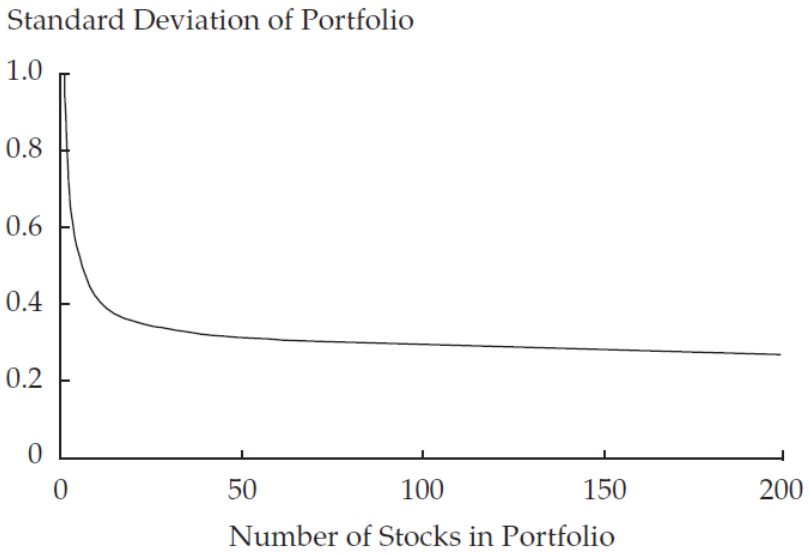

Figure 5-2. Expected Standard Deviation with Portfolio Diversification (all stocks have equal weight). The correlation between the returns of two stocks is 0.08, and the standard deviation of any stock is 1.0. (Statman 2004)

But an increasing number of portfolio positions increases the average transaction cost per position as well (Konno and Yamamoto 2003). Hence, to reach at the optimal number of portfolio positions, an investor should add stocks as long as the marginal transaction cost is lower than the marginal benefit, i.e. the reduction in risk by diversification (cf. Evans and Archer 1968; Statman 1987). This relationship has been subject to an extensive body of research. Evans and Archer (1968) found that adding more than 10 stocks to a portfolio cannot be economically justified (Evans and Archer 1968). In 2004, Statman revised this analyses and found based on current data that a portfolio can contain up to 300 positions before marginal cost outweigh marginal benefit (Statman 2004). Shawky and Smith (2005) analyzed U.S. domestic funds and found they hold a number of 40 to 120 positions and a positive correlation between fund size and number of positions (Shawky and Smith 2005).

Literature shows that the recommended number of positions increases during the last decades. Campbell et al. (2001) show that an investor needed 50 positions during 1986-1997 to reach an equal portfolio standard deviation than one could achieve in 1963-1985 with 20 positions. One reason is that the standard deviation between stocks dropped from 0.15 in 1984 to 0.08 in 1997 (Statman 2004). In addition, by the extended use of information technology financial markets became more efficient and hence transaction cost could be decreased (Hendershott, Jones, and Menkveld 2011). Obviously, spreading stock investments over industries is specifically important for diversification. But nevertheless, risk reduction is actually driven to a larger extent by

the number of positions than by diversification over industries (Domian, Louton, and Racine 2007; e.g. Statman 2004).

Considering this state of research, we include a range for the number of portfolio positions into our model. Thus, investors are able to express their view on the number of portfolio positions which is an integral part of an investment approach and hence should be included in an investment strategy. We denote the desired minimum number of portfolio positions with ***n1*** and the maximum number with ***n2***.

5.4.4 Stock Ranking Mechanism

At the heart of an investment strategy, and often even considered identical, is the decision which stocks to choose for a portfolio. When it comes to stock selection, there is a huge variety of approaches derived from different investment philosophies like fundamental investment, technical analysis and behavioral finance.

The question is: How can we integrate flexible support for such diverse approaches into our model while preserving its formal character to enable automatic execution of investment strategies? We embrace an approach suggested in Gottschlich & Hinz (2014), using a scoring mechanism to rank the stocks in the investment universe by preference. By incorporating the score, or metric, in the investment strategy specification, we include the knowledge or decision rule which stocks are preferable over others from the investor's viewpoint. Depending on the investor's view of the world, this could be a technical (e.g. momentum of previous week or moving average for last 200 days) or fundamental (e.g. price/earnings ratio) indicator or any other metric for which data can be provided. In fact, it is a function that states which stock is preferable over another based on selected criteria. Combination of metrics, e.g. by building a weighted sum of several scores, allows for more complex ranking mechanisms. Investors can create a library as a directed graph of metrics which they can employ in their investment strategies. This graph of metrics represents the investor's knowledge and analytical capability for stock selection.

A system implementing our model would compute the score(s) when a portfolio layout needs to be determined (e.g. at a rebalancing event) and rank all stocks according to their score (cf. Figure 5-1). For example, a possible (simple) metric would be the price/earnings ratio. The system would then compute this metric for every stock at a specific day (when a rebalancing occurs) and sort all stocks by this value such that stocks with low price/earnings ratios are ranked better than those with higher ratios

(assuming a long strategy)[14] and hence preferred for portfolio selection. Ranking can also comprise filtering, i.e. stocks which should not be taken into account receive prohibitive scores. In the example above, stocks with price/earnings ratios of e.g. above 50 might be excluded.

By introducing a metric for stock order preference in our model, we can still provide an abstract formal model description, but allow for flexible specification of stock preference in an application scenario. For an application example, see section 5.5. We denote the stock preference score in our model with ***s***.

5.4.5 Distribution of Capital

To arrive at a final portfolio layout from a list of ranked stocks, we need to decide how to split the available capital among those stocks. How many stocks starting at the top of the list should be considered and which amount of money should be distributed to each of them? While the first part of the question can be answered with the help of the model components ***n1*** and ***n2***, specifying the desired range of portfolio positions, the question of how the capital should be split among selected stocks forms another dimension of investment decision.

5.4.5.1 Naïve Diversification (1/N)

The easiest way to split a certain amount of capital to N selected stocks is an equal distribution. With this method, at every rebalancing event, each portfolio position is adjusted to the same share of total capital C (which is C/N) (cf. DeMiguel, Garlappi, and Uppal 2007; Tu and Zhou 2011). Advantages of this approach are an easy implementation and independence from biases potentially caused by estimation or optimization techniques for returns or weights. Also winning and losing positions can be identified at a glance. Practitioners do use such easy methods of capital spread for their investments and they are able to compete with more sophisticated methods (DeMiguel, Garlappi, and Uppal 2007).

5.4.5.2 Portfolio Optimization (Markowitz)

Given a set of stocks the portfolio selection approach as proposed by Markowitz (1952) determines, based on their historic development, weights for each of the stocks to derive an efficient portfolio. Efficient means that there is no other stock selection that beats the given solution in terms of risk/return ratio. Technically, the ap-

[14] This is a simplified example for illustration purposes which does not necessarily implement a very successful strategy.

proach minimizes variance (= risk) for a given return level or maximizes return for a given risk level. The approach received criticism, partly due to the fact that it maximizes errors in the estimation of expected return or stock correlation. Those estimates are necessary for the method to determine position weights (Chopra and Ziemba 1993). In addition, portfolios built using Markowitz' optimization approach did not necessarily outperform other methods of portfolio construction (cf. Tu and Zhou 2011). But nevertheless, it is a reasonable approach to determine an optimal portfolio regarding risk/return trade-off. For our purposes, having an already pre-ranked list of stocks and trying to determine how much capital should go into which position, the method is a convenient way to determine optimal weights based on risk/return figures.

Apart from the two capital distribution methods mentioned, investors can specify their own methods, too. As we have a ranked list of stocks, another reasonable distribution method would be e.g. a diminishing distribution, assigning most weight to the best-ranked stock and reducing the share while descending the list. This enables investors to reap the benefits of the ranking to a greater extent than with naïve diversification. It is obvious that the decision on how to distribute the available capital on the stocks selected has influence on the portfolio performance outcome and should be included in our investment strategy model.

We model the capital distribution method as a function that receives the ranked list of stocks and returns a number of weights for those stocks. For naïve diversification, the function could simply take the top 10 stocks and assign a weight of 10% to each of them. The optimization approach after Markowitz is more complex and involves running an optimization algorithm. Other approaches are possible, by implementing an appropriate function and including it in the investment strategy model. We denote such a function of capital distribution with ***d***.

5.4.6 Model specification

Based on an extensive body of investment literature, we identified several components (cf. Table 5-1) which affect portfolio performance and hence are determinants for portfolio creation that should be included in a formal model for investment strategies. Formally, we specify an investment strategy ***IS*** as an instance of the following tuple of model components:

$$IS = \langle r, bf, bt, n1, n2, s, d \rangle$$

Table 5-1 summarizes the model components. This model provides a way to specify investment strategies in a way machines can interpret and execute and hence is an

important step towards an automated portfolio management system (e.g. Gottschlich and Hinz 2014). In the upcoming Evaluation section, we implement the model and show how different values for strategy parameters affect portfolio performance, thereby showing the effectiveness of the model.

Table 5-1. Model Components

Component	Description
IS	Investment strategy
r	Investor's risk tolerance
bf	Rebalancing frequency
bt	Rebalancing threshold
n1	Minimum number of portfolio positions
n2	Maximum number of portfolio positions
s	Stock ranking metric (score)
d	Capital distribution method

5.5 Model Evaluation

In this section, we want to show how the model is applied to formulate an investment strategy which can be executed by a system to yield a target portfolio layout based on the strategy parameters. To show that our model provides the intended functionality, we introduce a base scenario and a number of test scenarios which differ in one of the model parameters to isolate the effect of the specific parameter. We compare the resulting portfolio of each test scenario with the base scenario to observe the results of the parameter adjustment. The purpose of this evaluation is not to identify the most profitable parameter settings, but rather show that changes of parameters have an effect on portfolio performance and hence there inclusion in the model is justified. The search for optimal parameter settings is subject to further research which finds strong support by an automated laboratory environment based on our formal strategy model.

5.5.1 Implementation

To evaluate the functionality of our model, we implemented a prototype system to execute an investment strategy which is specified using our model. The system takes a specified investment strategy and executes it over a specified time period while tracking the portfolio layout and performance development. In fact, the system converts the specified investment strategy into a portfolio layout for a specific day. Besides the portfolio, the system tracks the amount of cash available for a simulation run. We conduct our simulation with an initial cash value of 100,000 EUR.

We implemented the prototype in JAVA using the Spring Framework and a MySQL database to store stock quotes, investment strategies and portfolio layouts. To compute the complex Markowitz calculations, we integrated the statistical software R which is executed and controlled by the system.

We ran all the scenarios in a time period of two years: January 2009 to December 2010. Stock quotes were closing prices at Frankfurt Stock Exchange. For transaction cost, we used the cost model of a large German retail broker who charges 4.90 EUR per transaction plus 0.25% of transaction volume; minimum fee is 9.90 EUR and maximum fee is 59.90 EUR (comdirect Bank AG 2013). Transaction costs were aggregated to an external account and hence had no effect on the available investment capital. Further, we did not consider payment of dividends or taxes. One scenario run took approximately between 40 and 70 minutes on an Intel Core 2 Duo with 2.53 GHz and 4 GB RAM.

5.5.2 Scenarios

We defined 6 scenarios for evaluation purposes (see Table 5-2). The first scenario serves as a base scenario. The other scenarios each vary one parameter of the model to show the resulting effect on portfolio performance. By varying only one parameter at a time, we ensure that observed effects were caused by the specific manipulated parameter. Thus, we can evaluate the effectiveness of the model parameters. We keep one parameter fixed: stock selection. For stock selection, we use the score from Gottschlich and Hinz (2014) (GH). This score is based on a collective estimate of a large crowd on a stock voting platform and measures the potential price increase or decrease that the crowd assigns to a certain security on a certain day. That means for a long strategy (which we apply here), we use this score in a descending order to rank the stocks with the highest potential first (for details cf. Gottschlich and Hinz 2014). We do not change the stock selection parameter throughout the test run, because it is not our focus to compare stock selection mechanisms. We could use any other score as well with its respective results on portfolio performance to evaluate the effects of the other parameters.

For capital distribution, we use the Markowitz portfolio selection theory (PST) approach to arrive at a portfolio with optimized risk/return profile in two variants: PST(12) uses price history of the last 12 months to compute the variance of a portfolio position, while PST(6) only uses past 6 months. Thus, PST(12) should be more stable and react slower to changes of volatility in a security's development, while PST(6) reacts quicker, but also more volatile.

Table 5-2. Model Evaluation Scenarios

Scenario	Risk Tolerance *r*	Rebal-ancing Frequency *bf*	Rebal-ancing Threshold *bt*	Min. # Positions *n1*	Max. # Positions *n2*	Stock selection score *s*	Capital Distri-bution *d*
S0	30%	weekly	0%	0	10	GH	PST(12)
S1	60%	weekly	0%	0	10	GH	PST(12)
S2	30%	monthly	0%	0	10	GH	PST(12)
S3	30%	weekly	10%	0	10	GH	PST(12)
S4	30%	weekly	0%	0	2	GH	PST(12)
S5	30%	weekly	0%	0	10	GH	PST(6)

5.5.3 Results

As an external benchmark for performance comparison, we show the development of the DAX stock index which captures the 30 largest German companies (based on market capitalization and turnover). Table 5-3 shows an overview of all scenario portfolio results, while Figure 5-3 shows a plot of the scenario portfolio performances over the whole period. At a first glance, we see that the performances of the different scenarios differ, giving a first indication that the parameters included in the model are indeed determinants of portfolio performance and hence should be contained in our model. An exception is the result of Scenario S3 which performs identical to S0. We will discuss this observation in detail in the subsection of Scenario S3.

5.5.3.1 Scenario S0 – Base scenario

The base scenario (cf. Table 5-2) applies a rather conservative risk tolerance of 30% with a weekly rebalancing frequency. The rebalancing threshold is 0% which means that every deviation from the position target weights leads to an adjustment of portfolio position size. We specify no required minimum of portfolio positions, letting the system decide to invest or keep cash when a rebalancing event occurs. For this test run, we want to maintain a simple portfolio and hence set the maximum number of positions to 10. In formal terms, S0 can be specified as:

$$S0 = <0.3, Weekly, 0\%, 0, 10, GH, PST(12)>$$

Looking at the results (Table 5-3 or Figure 5-3, respectively), we see that the DAX benchmark develops positively with a return of approx. 40%, while S0 clearly outperforms the DAX benchmark with a return after transaction costs (TC) of app. 126%. These are the absolute results, but for our subject, we are more interested in the relative results between scenarios.

Table 5-3. Overview on the Model Evaluation Results (rounded)

Initial Capital	Resulting Capital	Return Rate (rounded)	Transaction Costs (TC)	Result – TC	Return Rate – TC (rounded)	Risk
Dax Benchmark						
100,000€	140,551.81€	40.44%	n/a	n/a	n/a	13.92%
Base Scenario S0						
100,000€	241,836.86€	141.84%	16,103.97€	225,732.89€	125.73%	27.25%
Evaluation Scenario S1						
100,000€	300,544.64€	200.54%	17,187.04€	283,357.6€	183.36%	35.74%
Evaluation Scenario S2						
100,000€	181,097.76€	81.10%	4,825.14€	176,272.62€	76.27%	18.46%
Evaluation Scenario S3						
100,000€	241,836.86€	141.84%	16,103.97€	225,732.89€	125.73%	27.25%
Evaluation Scenario S4						
100,000€	207,683.79€	107.68%	7,293.08€	200,390.7€	100.39%	19.30%
Evaluation Scenario S5						
100,000€	370,286.18€	270.29%	21,877.22€	348,408.96€	248.41%	33.01%

5.5.3.2 Scenario S1 – Risk tolerance

In scenario S1, we change the investor's risk tolerance to 60%. As we expect a higher risk to yield a higher return (Ghysels, Santa-Clara, and Valkanov 2005), the Scenario S1 should outperform the base scenario. Formally, we specify S1 as:

$$S1 = \langle 0.6, Weekly, 0\%, 0, 10, GH, PST(12) \rangle$$

From Table 5-3, we see that the S1 portfolio indeed outperforms the S0 portfolio by almost 60 points while the risk associated with the portfolio also increased to 35.74%. As we altered no other parameter except the risk tolerance, we conclude that the increased risk tolerance led indeed to a higher portfolio performance at the price of a higher risk. These observations confirm the results found in literature (Ghysels, Santa-Clara, and Valkanov 2005; Guo and Whitelaw 2006) and show the functionality of this parameter in our model.

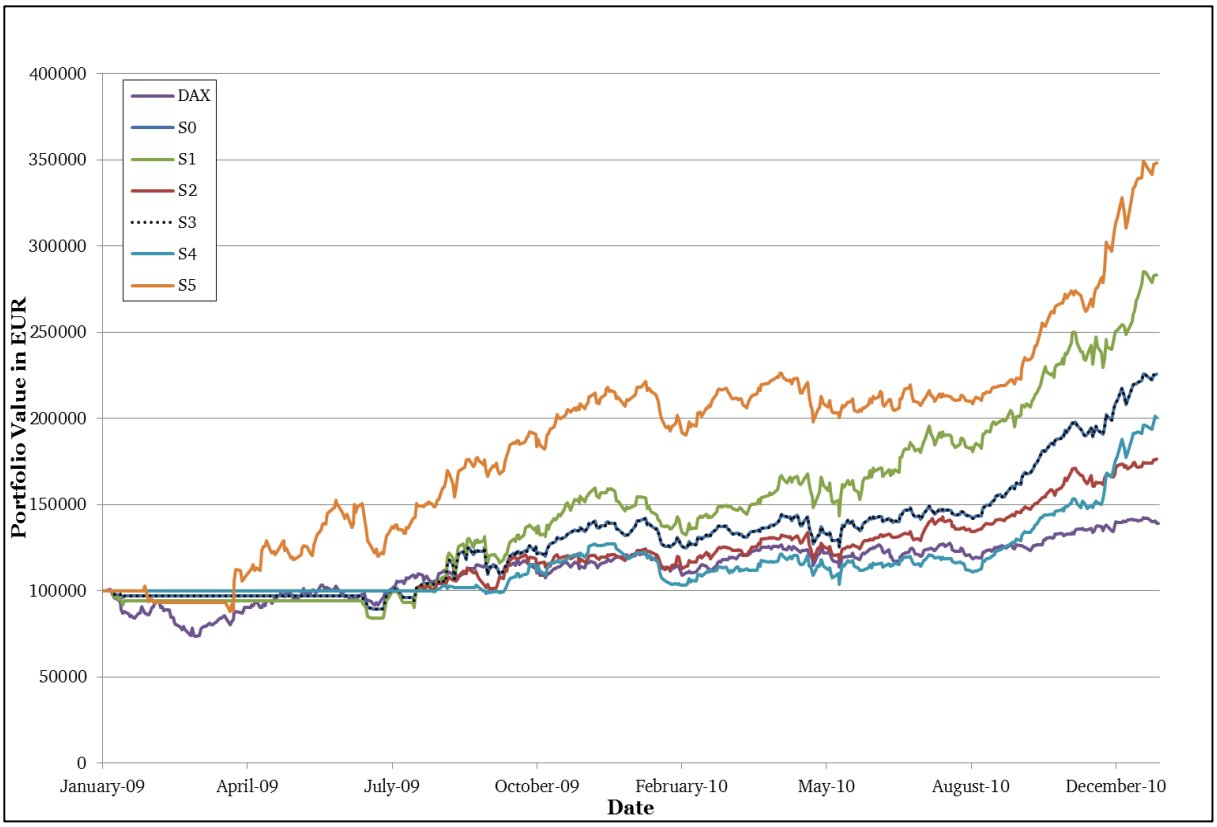

Figure 5-3. Performance of Scenario Portfolios with Transaction Costs Deducted

5.5.3.3 Scenario S2 – Rebalancing Frequency

In this scenario, we change the rebalancing frequency from a weekly to a monthly portfolio check and adjustment. Since Buetow et al. (2002) showed that a smaller rebalancing intervals increase returns, we expect a negative impact of this parameter change compared to the base scenario S0. The full specification of S2 is:

$$S2 = \langle 0.3, Monthly, 0\%, 0, 10, GH, PST(12) \rangle$$

Table 5-3 shows a performance for the S3 portfolio of 81.10% compared to 141.84% of the base scenario S0. So changing the rebalancing interval alone from weekly to monthly and keeping everything else equal, the performance drops by app. 60 points. In addition, due to the less frequent rebalancing interval, there are less trades to be made (123 trades in contrast to 442 trades in the base scenario), resulting in lower transaction costs, as Figure 5-4 shows. However, the lower transaction costs cannot compensate the loss in price development. All in all, these findings show that the rebalancing interval is an important determinant of portfolio performance and a necessary part of an investment strategy specification.

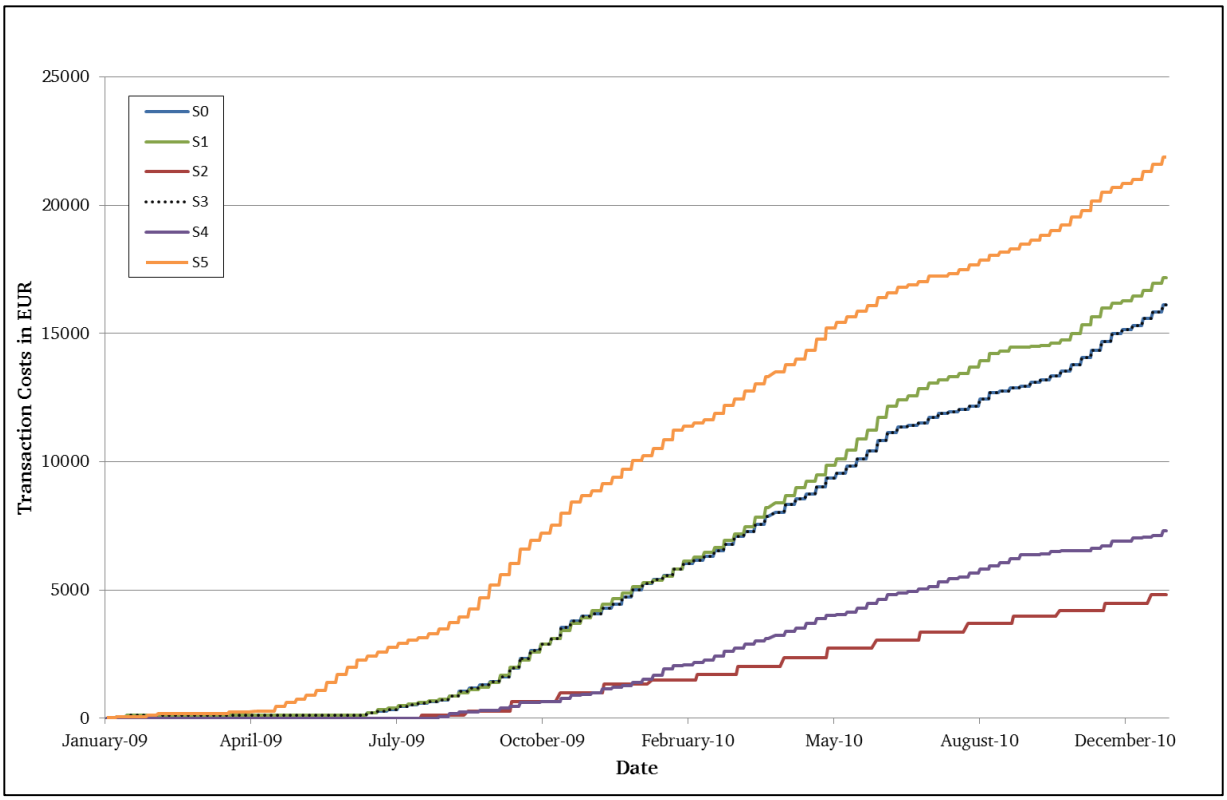

Figure 5-4. Development of Transaction Costs

5.5.3.4 Scenario S3 – Rebalancing Threshold

The rebalancing threshold defines the maximum deviation a portfolio position may show against the target weights before its size is re-adjusted. Buetow et al. (2002) reported a positive impact by increasing the rebalancing threshold from 0% to 10%. Another positive effect could arrive from lower transaction costs, as a higher tolerance towards target weight deviation can lead to a lower number of trades and hence reduce transaction costs. The formal specification of S3 is as follows:

$$S3 = \langle 0.3, Weekly, 10\%, 0, 10, GH, PST(12) \rangle$$

The results of Scenario S3 are identical to those of the base scenario S0. Why is that? This is due to a conflict of parameters: the applied metric for stock selection (GH) is very volatile in its recommendations leading to a very different list for every rebalancing event. So a rebalancing based on this metric is rather fundamental exchange of portfolio positions. Because the target weights of portfolio positions and the positions themselves change so much, this parameter is masked by the stock selection metric and shows no effect in the current test run. Future evaluations of the model should analyze if this parameter is effective with different evaluation data.

5.5.3.5 Scenario S4 – Maximum number of portfolio positions

Because the number of portfolio positions affects diversification of a portfolio which is connected to portfolio performance and risk, we adjust the maximum number of portfolio positions in Scenario S4 from 10 to 2 and evaluate the effect. Due to the lower diversification, we would expect a higher risk associated with the portfolio. As we decrease portfolio size by a large extent, we also expect transaction costs to be lower than in the base scenario. Scenario S4 is specified as follows:

$$S4 = \langle 0.3, Weekly, 0\%, 0, 2, GH, PST(12) \rangle$$

S4 has a return rate before TC of 107.68% compared to 141.84% in the base scenario (cf. Table 5-3). Surprisingly, the risk is not increased compared to the base scenario, but instead dropped to a value of 19.30% compared to 27.25% in the base scenario. This is against the expectations from previous literature, which predict a higher risk with less diversification. We conclude that this is a random effect with our evaluation data set. However, in accordance with previous research is the drop in performance compared to the base scenario which comes with the reduction of risk. In this respect, our results are consistent with literature.

As expected, the transaction costs also drop from 16,104 EUR in the base scenario to 7,293 EUR (cf. Table 5-3 and Figure 5-4) – less than half. However, these savings cannot over-compensate the loss caused by lower diversification.

5.5.3.6 Scenario S5 – Capital distribution

For the last evaluation scenario S5, we modify the method of capital distribution. In all previous scenarios, including the base scenario, we took a history window of 12 months to compute the volatility and correlation metrics for stocks which are needed for the Markowitz portfolio selection. Now we shorten this window to 6 months. By doing so, the investment behavior of the system should be more responsive to recent market developments and act more agile on market changes. Formally, we specify S5 as:

$$S5 = \langle 0.3, Weekly, 0\%, 0, 10, GH, PST(6) \rangle$$

Scenario S5 shows the strongest performance of all portfolios – 270.29% return compared to 141.84% in the base scenario. Looking at Figure 5-3, we see that all other scenarios show no trading activity during the first few months of the evaluation period. The reason is a rather volatile bear market in 2008, which ended in 2009 and turned into an upwards trend. The scenarios which use the past 12 months to estimate stock risk, stick longer to the negative evaluation of stocks before the positive

developments allow the system to invest again instead of keeping a 100% cash position. With a 6-month time window for risk assessment, the positive market development leads to a quicker pick-up of the bull market by Scenario S5 and hence explains its superior performance compared to the other scenarios.

We confirm that the capital distribution mechanism is a crucial part of an investment strategy and even slight modification can have large impact on portfolio performance. Hence, we are confident that the capital distribution method should be an integral part of our formal model for investment strategies.

5.6 Conclusion

In this paper, we introduced a formal model to specify investment strategies in a generic way. Based on an extensive review of current investment literature, we identified determinants of stock portfolio performance and formalized them as components in our model. To the best of our knowledge, no such universal and integrated approach of formalizing investment strategies existed before. However, this is a crucial ingredient to bridge the gap between pure decision support systems, which support a human decision maker who then executes the decision and fully automated trading systems that are closed (and often secret) systems and do not allow for an interactive exploration of strategies by an investor. In contrast, our model serves as a language to express investment strategies by investors and still enable execution to derive a target portfolio layout automatically when implemented in an appropriate portfolio management system (cf. Gottschlich and Hinz 2014). This fundamental conceptual framework serves both researchers and practitioners by providing a generic laboratory environment to model and analyze new investment ideas and test them in a comparable way with strong automation support. By providing a “language” to describe an investment strategy formally, we also create a precise way to express, communicate and store investment approaches. Further, a portfolio management system which implements the model provides generic support for a wide range of investment approaches which are applicable to automatic portfolio management. Alternatively, such a system provides decision support up to the derivation of stock orders which can still be controlled by human investors before execution.

While we provide a first version of an integrated investment strategy model, there is still room for improvement. First, there might be (and certainly are) other factors which affect portfolio performance which we have not yet addressed with our model. But we are confident that our suggestion in this paper already covers the most widely accepted and most common components of investment strategies. In terms of risk

modeling, Value-at-Risk as a common approach used by practitioners should be included in the implementation of the model to make it more applicable for practical use.

Second, the results of our evaluation were not final with respect to the effect of the rebalancing threshold (Scenario S3) and the effect of restricting the number of portfolio positions (Scenario S4). While the effects of these parameters are well founded in theory, based on our data set, we were not able to find support for this theory. Further improvement efforts of the model should include a re-evaluation of these parameters to see if their presence in the model is justified. A third venue for improvement is the handling of conflicts between parameters – as we discovered in Scenario S3, when the stock ranking mechanism made the rebalancing threshold ineffective. The investor could specify a priority of parameters in case of conflict to overcome this limitation.

But most exciting future uses will probably comprise a massive sensitivity testing of investment strategies to apply statistical methods on the significance of strategy parameters. By executing a large number of slightly different parameter settings and analyze their effect on portfolio performance by applying statistical methods, new levers for successful investment approaches could be identified and evaluated. This is an exciting outlook for both, practitioners as well as researchers, as our model would provide the fundamental building block for such a batch testing of investment approaches in a laboratory environment or for real money investments.

In conclusion, we are confident that our proposition is a valid and valuable approach to enable a formal specification of a wide range of investment strategy approaches. We showed by our prototypical implementation that such specified strategies can be executed by an appropriate portfolio management system and that our model is suitable to narrow the gap between pure decision support systems and automated trading systems.

6 A Cloud computing Broker Model for IaaS resources

Title	A Cloud computing Broker Model for IaaS resources
Author(s)	Gottschlich, Jörg, Technische Universität Darmstadt, Germany Hiemer, Johannes, Technische Universität Darmstadt, Germany Hinz, Oliver, Technische Universität Darmstadt, Germany
Published in	*Proceedings of the European Conference on Information Systems*

Abstract

Infrastructure-as-a-Service (IaaS) is the most flexible form of cloud computing and provides great opportunities to acquire and release computing resources as necessary. However, consumers face an increasingly opaque market due to growing number of providers and tariff options. As an approach to aid consumers finding the right tariff matching their needs, we suggest a broker model as an intermediate between consumer and providers. Based on a demand specified by the consumer our broker model implements a process to identify the most cost-efficient tariff. Using a database with performance metrics we collected by benchmarking the provider machines, the tariff selection is based on true performance data and is able to consider task specific component dependencies, e.g. for CPU intensive tasks. We demonstrate the functionality of our model with a test run.

Keywords: Cloud Computing, Infrastructure-as-a-Service, Broker, Sourcing

6.1 Introduction

With the ongoing growth of the Cloud Computing market, many believe IT to become the 5th form of utility, as ubiquitous and easy to source like water, electricity, gas and telephony (Buyya et al. 2009). Cloud Computing comprises three layers of services (Figure 6-1): Software as a service (SaaS) delivers end-user applications for specific tasks (e.g. expense reporting) or business areas (e.g. Customer relationship management). Platform as a Service (PaaS) offers execution environments for programming languages like Java, .NET and JavaScript. Finally, Infrastructure as a service (IaaS) as the most fundamental service type provides runtime environments for virtual machines (VMs). Providers of IaaS offer a simulated hardware environment that customers can use to build their very own software stack, down from the operating system up to installed applications. Due to their virtual nature, it is easy to copy and move VMs, thus providing a great flexibility to change the hosting environment. These characteristics turn IaaS into the most universal type of Cloud Computing, providing "pure" computing power and promise a highly competitive market environment as switching costs are lower compared to "hardware" hosting. The focus of this paper is at the IaaS service level of Cloud Computing.

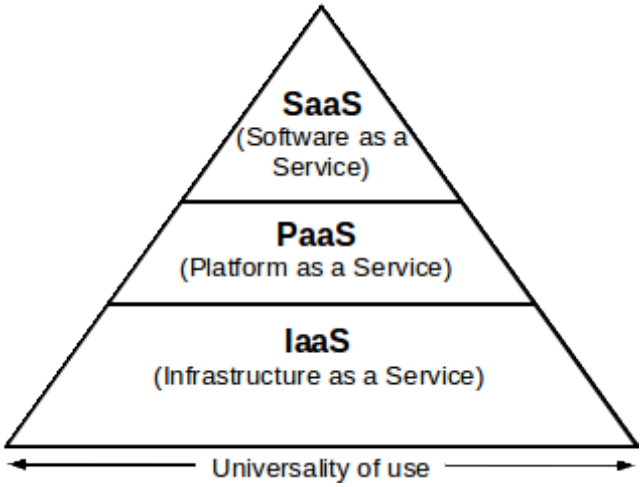

Figure 6-1. Cloud Computing service layers

As the market for Cloud Computing resources grows (Berlecon Research GmbH 2010), not only the amount of available providers[15], also the complexity of service offers and pricing models – and hence market opacity – increases. This is partly due to the complexity of the product, but experience from other markets show, that it can as

[15] e.g. http://vcloud.vmware.com/vcloud-ecosystem#view=full

well be intended by the providers to hinder price comparison (Frischmann, Hinz, and Skiera 2012). When sourcing IaaS services, resource consumers who want to identify the most efficient provider for their resource demand, need to compare offers based on different subscription (e.g. pay-per-use, monthly subscription) and pricing models (e.g. fixed fee per hour, utilization-based pricing) or configuration and service levels (e.g. resource configuration of VMs in terms of CPU power or RAM size, availability levels etc.). Due to the virtual nature of the systems, it is not straightforward to estimate the performance of a Cloud Computing instance. Certainly, providers advertise specific configurations, e.g. number of virtual CPUs (vCPU), RAM size and HDD size – but the performance of one "virtual CPU" might be quite different between two provider tariffs; also RAM and HDD performance can be different due to technical infrastructure or intended tariff limitations (Cloudspectator 2012).

In addition to the system performance opacity, customers have very different requirements for their computational tasks. For example, database applications have high demands on storage performance, but less on CPU compared to tasks like movie transcoding or cryptography, leading to changing priorities on system component performance for different customers. If those priorities are stated explicitly per component, we receive a load profile which can be used to measure and distinguish tasks by their component utilization (see section 6.3.1 for a formal definition). Thus, the broker improves market transparency for consumers of Cloud Computing resources by supporting them to find a supplier matching their computing demands. Additionally, the consumer is able to filter for qualitative criteria (minimum component sizes, service level, geographic location etc.) to identify tariffs within desired boundaries. To validate our proposition, we collected tariff and performance data for 14 provider tariffs and applied our model in a proof-of-concept scenario.

The following section 6.2 provides an overview of related work in the area of Cloud Computing resource brokerage. After that, we introduce our approach and formalize the model we propose in section 6.3. Subsequently, we explain the data collection process (section 6.4.1) and the broker process that identifies a matching provider for a given resource demand (section 6.4.2). The approach is validated with real world data in section 6.5, before we conclude our work in the final section 6.6, pointing out limitations and opportunities for future work.

6.2 Related Work

Research of optimal resource distribution in cloud environments covers many different aspects of distribution approaches. The different approaches focus on different

service layers of Cloud Computing and different optimization objectives. Some approaches only cover the technical distribution, while others try to address both technical and economic aspects. Extracting the work relevant for this paper, we focus on optimization approaches covering the IaaS layer and analyzing cost aspects and/or performance measurement (benchmarking).

In an early work, Binnig et al. (2009) discuss the possibilities of benchmarking resources in the Cloud. The motivation behind their analysis is the assumption that classical benchmarking strategies are not applicable anymore in Cloud environments because of changing computing environments. As a consequence Binnig et al. suggest a framework, which traces Cloud resources for the adaption of different usage scenarios from the perspective of load changes, performance and costs. Their approach primarily focuses on the application in a PaaS environment, as the presented automatic scaling mechanisms are not applicable to an IaaS environment.

Although derived from Grid Computing, Chard et al. (2010) analyze the architecture of a broker model for the optimal distribution of VMs in IaaS and Grid environments. In their paper, VMs are defined as *distributed resources.* The paper analyses different allocation models, but neglects different types of VMs and Quality-of-Service aspects. The missing specifications of the resources make an adaption of their broker model very difficult to today's Cloud Computing market.

Cooper et al. (2010) address the benchmarking and scaling mechanisms in cloud environments. In contrast to the previously presented related work, they only focus on database systems like Cassandra, HBase, Yahoo!'s PNUTS, and a simple distributed MySQL implementation. Because instances of a database in a cloud follow a completely different architecture when compared to plain IaaS instances and are more similar to PaaS, their benchmarking focuses on performance, scaling, availability and replication. Although the authors show the different characteristics and behavior of their testing environment, the results cannot be used as a basis for a broker model in IaaS environments, as they build upon a completely different architecture.

Iosup et al. (2011) cover the usage of Cloud Computing distribution for parallelized tasks in scientific computing. In their analysis, they compare four different cloud providers ranging from smaller ones to large global cloud provider like Amazon. In their setup they define different types of tasks that are executed for a specified range of time. These task-sets run on the infrastructure of the four cloud providers. The results show that outsourcing scientific computing to Cloud Computing providers can lead to significant cost cutting and improved performance. In comparison to the work at hand, their approach is a valid solution for task driven demands. These demands

could even be applied with spot instances or pay-per-use subscription models. The presented approach assumes tasks can be stopped and resumed at any time and thus is not suitable as a general approach for IaaS resources.

Li et al. (2010) compare public cloud providers. The work is based on CloudCmp, which is a systematic comparator for performance and costs of cloud providers. In their analysis they focused on four service types: elastic compute cluster, persistent storage, intra-cloud work and wide-area networks. For each of these service types the authors define a complete metric set, which is then applied to the benchmarking environment. Additionally, to show the applicability and the real world relevance, the paper conducts three case studies, each representing a typical load scenario for cloud driven applications.

El Kihal, Schlereth, and Skiera (2012) suggest two methods to make IaaS price comparison more transparent: hedonic pricing which decomposes the price into contributing values of the tariff's characteristics and a new-developed pricing plan comparison which aims to identify the most favorable requirement profile for each provider compared to its competitors. They validate their approaches in an empirical study, but only use the nominal performance descriptions given by providers and do not benchmark the true performance of the tariffs.

Finally, Garg et al. (2013) suggest the currently most complete discussion of a benchmarking and measurement environment for cloud providers. Their work is based on the service measurement index (SMI) by The Cloud Service Measurement Initiative Consortium (CSMIC) (2013), which is used to put up a Service Measurement Index Cloud framework SMICloud. SMICloud contains sixteen different aspects, also called key performance indicators, which are extracted from the SMI and which are highly relevant for computing in cloud environments. The paper however disregards important aspects like fixed prices, hidden pricing components and different tariffs for a single cloud provider.

6.3 IaaS Resource model

The goal of our broker model is to rank provider tariffs based on a given resource demand profile. To be able to do so, we need to collect pricing and performance data for each provider and make it comparable. Therefore, we introduce a formal model in this section that enables us to compute a comparable price for a given resource demand. Qualitative factors, e.g. the geographic location of the providers, can be specified by the consumer and serve as filters to restrict the selection to providers meeting those requirements.

We model a Cloud Computing instance for IaaS – a virtual machine (VM) – as a resource set consisting of three components: Number of virtual CPUs (#vCPU), amount of virtual random access memory in Gigabytes (RAM) and amount of storage capacity in Gigabytes (HDD). Formally, we denote such a resource set as vector $\vec{r}$ = (#vCPU, RAM, HDD). vCPU is no standard unit, providers use it to label different portions of computing power, e.g. real CPUs, CPU cores or a certain amount of Gigahertz. For RAM and HDD, usually only the *capacity* is transparent, but not the *performance* in terms of transfer rate which depends on the technology involved (e.g. different RAM frequencies or network attached storage vs. local drives).

The true performance behind those nominal resource descriptions differs between providers (Cloudspectator 2012). Hence, to enable an unbiased comparison of the providers' price-performance ratio, we need to normalize the prices of the providers to match the true performance they deliver with a computing instance. A broker, as an intermediate between demand and supply, is the ideal place to perform continuous monitoring and benchmarking of provider performance and use this information to redirect a user demand to the best matching offer. This enables consumers to get a more transparent picture of the true performance they receive.

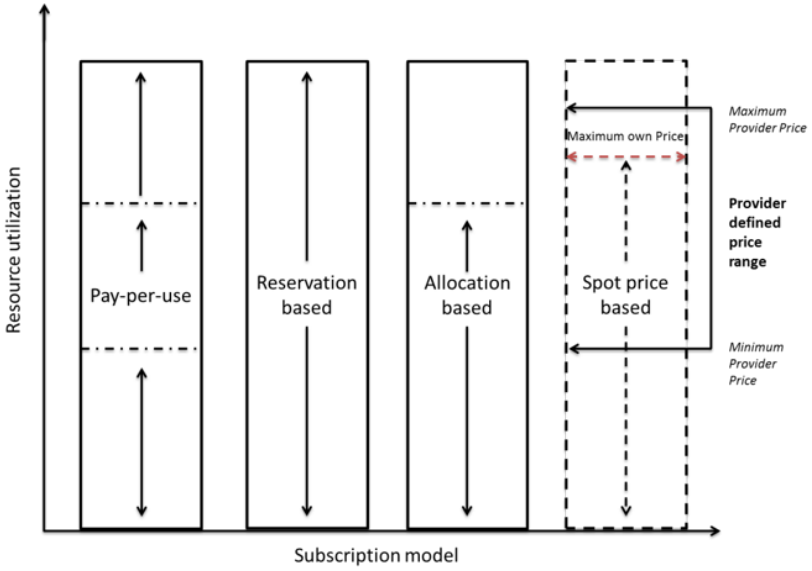

Figure 6-2. Subscription models for IaaS resources

IaaS providers show a variety of subscription models depending on the required service level and resource availability/reliability (cf. Figure 6-2): (1) with *reservation-based* pricing, consumers pay a fee (e.g. per month) for a fixed amount of reserved resources whether they use it or not. (2) *Pay-per-use* means consumers pay for resources only when and while they are actively used for computing. Some providers

also charge for the storage space when the VM is stopped, but not deleted. (3) *Allocation based* pricing is a hybrid model of reservation-based and pay-per-use: a certain amount of resources is reserved and subject to fixed payment. In addition, if peak usage should occur, some more resources can be utilized and are paid for the time they are occupied. (4) Finally, spot pricing is useful for non-urgent tasks which can be executed and resumed any time to make opportunistic use of available resources. The provider has the right to kill those instances anytime if other, higher prioritized, demands occur. This kind of tasks enables cloud providers to put even short periods of idle resource to productive use, increasing efficiency and utilization of their assets, without sacrificing any flexibility in capacity management.

Our model addresses qualitative factors (see 6.3.2 below) that play a role in the decision for or against a certain Cloud Computing provider, e.g. the geographic location of the servers, the software platform supported and so on. We use these factors to filter the list of provider tariffs and only those meeting the stated requirements make it to the consideration set of tariffs. Any data that is available about service providers can serve as filter criteria. Because the focus of this paper is on cost comparison of providers, we do not examine qualitative factors in detail. For an extensive overview on how to use qualitative factors in Cloud Computing brokerage, see Garg et al. (2013). For our model, the vector $\vec{c}$ contains the list of qualitative filter criteria which we apply on the list of providers.

6.3.1 Consumer – Resource demand

In our model, the consumer of Cloud Computing resources expresses a specific resource demand: $< \vec{c}, \vec{l}, T, t_{on}, s >$. The components of $\vec{c}$ provide qualitative criteria, which are used to filter the provider tariffs according to the consumer's requirements. Table 6-1 and Table 6-2 give a list of qualitative requirements, which can also be extended by further criteria at a later stage. Especially the minimum amounts of RAM and HDD capacity with a huge span depending on the application purpose are important criteria to exclude inappropriate tariffs. The load profile $\vec{l}$ contains the consumer's performance priorities for the components CPU, RAM and HDD. Depending on the intended use, application performance is more affected by CPU speed or RAM or HDD performance. For example, the load profile $\vec{l}$ = (60%, 30%, 10%) gives a weight of 60% to CPU performance, 30% to RAM performance and 10% to HDD performance. The components need to add up to 100%. This example setting indicates a rather computation-intensive task and hence a tariff providing high CPU at comparably low cost would be a better choice than one with high storage performance. In con-

trast, if we consider e.g. database applications, the opposite seems more preferable. Knowledge about task's core dimensions of resource usage increase the efficiency of the sourcing decision. Those providers will be preferred, who have low prices on the resource(s) the tasks uses heavily. By weighing the importance of each component in $\vec{l}$, the broker is able to make a more specific comparison of tariffs including strengths and weaknesses of providers' performance at component level (see section 6.4.1.2 for details on how this works). If the consumer does not provide a load profile, we can still apply a generic system benchmark to arrive at transparent prices with respect to overall system performance. But as component performance shows great variety among provider tariffs, we strongly recommend providing a load profile to get an efficient tariff recommendation.

To derive proper components weights for a load profile, we suggest using the following empirical approach on the specific task in focus: The component with the largest effect on task runtime is the most critical one; hence we should put most weight on the component showing the highest scalability effects for the task. In other words, we derive the component weight by the scalability of a component. Therefore, we conduct test runs of the task on different component configurations and measure the resulting total runtime. After a reasonable amount of test runs, we build a regression model on the task runtime as follows:

$$Runtime = \beta_0 + \beta_C * CPUPerf + \beta_R * RAMPerf + \beta_H * HDDPerf + \epsilon$$

CPUPerf, *RAMPerf* and *HDDPerf* are performance measures for each of the component (e.g. benchmark points) in all tested VM configurations (β_0 is the constant, ϵ the error term). By computing this regression model, we arrive at the runtime effects of CPU, RAM and HDD performance β_C, β_R and β_H which indicate the influence of each component performance on the task runtime. To make those coefficients comparable, we have to normalize every one of them using the value range of each measure (i.e. normalize β_C with the value range of *CPUPerf* ($=\beta'_C$), likewise for other components). In a final step, we standardize all three normalized measures to sum up to 100%: $\beta''_{\{C,R,H\}} = \frac{\beta'_{\{C,R,H\}}}{\beta'_C+\beta'_R+\beta'_H}$ and thus arrive at our load profile ($\beta''_C, \beta''_R, \beta''_H$).

While this approach might seem like a large overhead for a sourcing problem, considering high volume deployment scenarios with potentially long run times, it is a rather small one-time effort with potentially high savings. In addition, the learnings from such an empirical validation can be re-used for future scenarios and build up to a library of load profiles which can also be shared among users for standard tasks.

Due to the different subscription models (Figure 6-2), we need to take the expected runtime of the VM into account and unify all prices to hourly rates. We do so by introducing two parameters to our model: T denotes the total deployment time in hours, i.e. the time a VM is stored on a provider's system, regardless if it is switched on or off. t_{on} is the number of hours the VM is started and running ("on-time"). T-t_{on} is the remaining time share, when the VM is stored, but not started ("off-time", t_{off}). If the machine always runs and is never suspended then there is no off-time and $T = t_{on}$. The state of the VM also has implications on the prices charged by the provider. When the VM is running, it consumes all of the resource types in $\vec{r}$. While it is suspended, it consumes only storage capacity. Some providers have a separate price for this snapshot storage (e.g. Google 2013) which has to be taken into account for a total cost comparison. T and t_{on}, as defined above, have to be estimated by the consumer upfront based on the nature of the computation task and, over time, experience.

The last component s is an estimate of the HDD capacity required by the VM. It serves two purposes: 1) as a filter criterion to exclude tariffs not providing enough storage space and 2) to compute storage cost for off-time periods with a specific price for snapshot storage.

6.3.2 Provider tariffs

In our broker model, we operate on *provider tariffs* instead of *providers*. We do so to be able to handle different service tiers offered by one provider. If a provider has three service tiers and two of them do not meet the consumer's service requirements, there is still one left which does. So by using tariffs, we get a more detailed view on market supply and hence increase broker efficiency. The provider's overall attributes, e.g. location, legislation etc., are replicated to all tariffs, so they can still be used to set constraints. If one of those attributes violates a consumer constraint, all tariffs of the specific provider will be removed from the result set.

6.3.2.1 Pricing data

At the core of the provider tariffs is the pricing data. Due to different subscription models (Figure 6-2) and billing cycles, prices have to be harmonized. To do so, we rebase all prices to an hourly usage rate. In our model, CPU and RAM are charged only for the on-time *t*, while HDD at the size of *s* is charged for the whole deployment period *T*, for obvious reasons: we use the former resource only when the VM is running while the storage space is occupied no matter if the machine is running or suspended. Monthly prices are divided by 720 hours to make them comparable on a per-hour base. However, for the total cost calculation, billing cycles have to be taken into ac-

count to prevent wrong cost estimates. By adding a full month's charge right from the first hour if the tariff component has a monthly billing cycle, we ensure that the tariff only ranks high, if the total cost is still competitive compared to other tariffs (which means the intended runtime has to be long enough so that the monthly charge results in a low hourly rate). Other fixed upfront payments opposed by a certain tariff are added to the total cost as well.

6.3.2.2 Qualitative Criteria

We also collect qualitative criteria which are important in provider tariff selection and serve as filter constraints defined by the consumer. We distinguish between primary qualitative criteria and secondary qualitative criteria. The primary qualitative criteria comprise limitations regarding the resource components of a virtual machine. That means the consumer has the possibility to provide a set of conditions, containing the maximal and minimal constraints for CPU, RAM and HDD. Based on the experience the consumer has from his previous production or testing environment, these constraint assumptions should contain realistic and safe estimations of the maximal/minimal resource consumption.

Table 6-1. Primary qualitative constraints

Primary Qualitative Criteria	Name	Constraints
1	CPU	Lower and upper bound
2	RAM	Lower and upper bound
3	HDD	Lower and upper bound

The secondary qualitative criteria cover multiple aspects of Cloud Computing in general. The following list contains criteria such as location and legislation, support and customer service, backup and recovery aspects as well as technical details. An important but often disregarded aspect in the context of Cloud Computing is the location of the data center and applicable legislation of the operating cloud provider. Both do play a significant role for data security. While some countries like Germany have very strict data privacy laws, providers in other countries like the USA must offer data access for governmental institutions or secret services. This is not only problematic for the location of the data center itself, but also for the legal duties of the provider. This is only an excerpt of a much larger range of possible qualitative criteria. For an extensive list of qualitative factors in Cloud Computing brokerage, see for example Garg et al. (2013).

Table 6-2. Secondary qualitative constraints

Secondary Qualitative Criteria	Name	Filter values
1	Data Center Location	Country
2	Applicable Legislation	Country
3	Total availability time	Percentage
4	Support Time Ranges	Timespan
5	Recovery Time	Timespan
6	Minimal term of contract	Timespan
7	Offsite Backup	Frequency
8	Hypervisor Platform	Hyper-V, Xen Server, VMware, KVM

6.4 Broker model

The task of the broker as an intermediary is to match the resource demand defined by a prospective consumer's demand request with appropriate provider tariffs (Figure 6-3). We divide this task in two parts: (1) a data collection process, which collects the necessary data about the providers' tariffs and performance and (2) the broker process, which acts upon an incoming resource demand and identifies the provider that serves the demand most efficiently. After an initial data collection phase, these processes are independent and can be executed asynchronously. In the following two sections, we first describe the data collection part of the broker model before we explain the execution of the broker process.

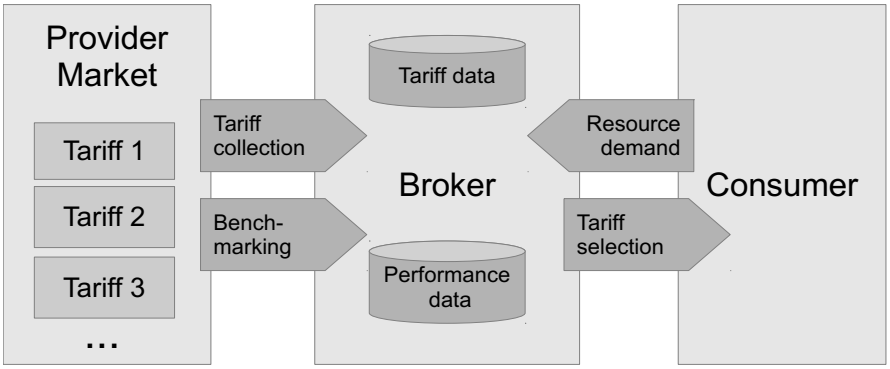

Figure 6-3. Broker as intermediary between consumer and provider market

6.4.1 Data collection process

The goal of the data collection process is to gather the necessary data to create market transparency. On the one hand, there is the tariff data, i.e. the information about prices and services published by the providers. The description of resources still follows a hardware analogy, but due to the virtual nature of Cloud Computing resources, the precise performance of virtual components is not transparently specified by the component description. Hence, we need additional performance data which we collect by continuously benchmarking the providers' infrastructure.

6.4.1.1 Tariff data

In order to make the tariff data available for performance comparison despite the diversity of pricing and subscription schemes, we convert the provider's raw data into the formal model outlined above. Currently, we do this manually, but at a later stage this process can be automated, e.g. by crawling the provider's web pages and convert the data automatically into the target model. Tariff data include at this stage: VM pricing per hour (on-time/off-time), number of CPUs per VM, amount of RAM per VM, amount of HDD capacity per VM

6.4.1.2 Performance data – Provider Benchmarking

To invoke transparency about the true provider performance i.ndependent from their announced resource descriptions, we introduce a benchmarking component into our model. For every provider tariff, we order an instance and use it to run benchmarks on the system and the component (CPU, RAM and HDD) performance. For the brokerage process, we use the benchmark results to standardize the prices in terms of performance. We describe the data collection of the performance data in this section and refer to the section 6.4.3 for details on how we compute the price-performance ratios used for tariff selection.

To measure the performance of an IaaS provider many different benchmarking suites exist. One of these benchmarking suites is UnixBench[16]. UnixBench was created in 1983 at the Monash University. It was later taken over and expanded by Byte Magazine. For our benchmarking, we choose UnixBench, because it is well known for its reliable results and its ability to not only cover single CPU systems, but also multi CPU systems, both penetrated with single- and multi-tasks. The UnixBench testing environment has nine different testing scenarios of which we pick three. They are listed and in Table 6-3 along with the respective component they test.

Table 6-3. Overview of UnixBench components

Name	Description	Component
Execl Throughput	This test measures the number of execl calls that can be performed per second.	CPU
File Copy	This measures the rate at which data can be transferred from one file to another, using various buffer sizes.	HDD
Process Creation	Process creation refers to actually creating process control blocks and memory allocations for new processes, so this applies directly to memory bandwidth.	RAM

[16] https://code.google.com/p/byte-unixbench/

In addition, UnixBench provides a system performance index which measures overall system performance. We will use this system benchmark score to compare tariffs if the consumer specifies no load profile. The result of this comparison is a general list of provider tariffs ranked by price-performance ratios, but it can still be filtered by various qualitative constraints.

By frequently running a benchmark for each component x of the resource vector $\vec{r} =$ (CPU, RAM, HDD) for each provider tariff i, we receive a time series of benchmark values $X_i(t)$. At this first approach, we compute a simple moving average $\bar{X}_i$ from this series and use it as performance benchmark value for component x of provider tariff i. At a later stage, using historic performance data, it might be possible to detect patterns of low or high performance per provider and adapt to those in the selection of providers.

6.4.2 Broker process

Upon a consumer's resource demand request the broker process returns a list of capable provider tariffs sorted by cost-performance ratio. We apply a 4-step-process based on the formal model introduced before. Using the formalization of the problem, we are able to implement the process and execute it automatically, e.g. as a web service. Companies can thus access this web service and use it for their sourcing decision in real-time based on recent price and performance data. Figure 6-4 gives an overview of the process which we describe in this section. The process kicks off when a consumer sends a resource demand request $< \vec{c}, \vec{l}, T, t_{on}, s >$ which includes tariff constraints, a load profile, total deployment time, on-time and HDD size. First, the broker uses the consumer's constraints to exclude tariffs which do not meet the requirements. For the remaining tariffs, we use the load profile to compute an individual cost-performance ratio based on the component benchmarks. If the consumer does not provide a load profile, we compute a general cost-performance ratio based on a system benchmark.

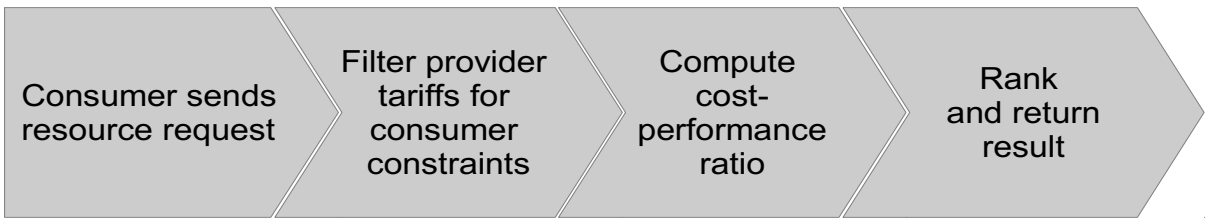

Figure 6-4. Broker process

6.4.3 Compute cost-performance ratio

There are two levels of price-performance ratio computation: The first (and simple) one is independent of a consumer's load profile and uses a generic system benchmark

to normalize prices. This gives an unspecific indication about the "general-purpose" performance of the VM and hence is a good approximation if no more specific information about the task nature is available, i.e. if the consumer does not specify a load profile. The second (and more complex) level of price-performance calculation considers the priority of components as indicated by the consumer load profile $\vec{l}$, which is used to weight benchmark results on a component level. This leads to a tariff ranking which is biased towards the application scenario stated by the consumer and hence prefers those tariffs, where the price-performance relations are in favor of the specified main components. We first explain the general price-performance ratio calculation and then the calculation based on the load profile.

6.4.3.1 General system price-performance ratio

Table 6-4 illustrates an example of price-performance calculation without a load profile. For each tariff, we have the system benchmark value in the benchmark's output unit (benchmark points, BP). We divide the announced price of each tariff by the benchmark result to get the price per BP (cf. Table 6-4). This is a very simple method to get a relative price depending on the system performance. Looking at the price per BP, we see that provider tariff B is actually cheaper than provider tariff A with respect to the delivered performance – this would not have been transparent from the announced price. All in all, we see that with no further constraints, tariff C provides the lowest price for 1 BP and hence the highest performance for a given price.

Table 6-4. Numerical example for the overall system performance-ratio calculation

Provider tariff	Benchmark result $\overline{X}_i$ (Benchmark Points, BP)	Announced price ($/h)	Price per BP ($/h / BP)
A	78.1	0.02	0.0002561
B	329.8	0.06	0.0001819
C	3088.0	0.48	0.0001554

6.4.3.2 Application-specific price-performance ratio

If the consumer provides a load profile $\vec{l}$, which describes the relative importance of components, we can use the additional information to find a tariff that is efficient for the specific usage scenario. Table 6-5 illustrates this process if we assume a load profile of (50%, 10%, 40%). First, we divide the price into components according to the load profile to reflect the weight a user assigns to a component. By using the price to distribute weights, we avoid the need to make assumptions about the relation of benchmarking values between components. Afterwards, we divide that component specific price by the component performance we measure for each tariff. When we add up those components, we arrive at a user-specific price in $/h/BP. For an example see Table 6-5: The price of 0.02 $/h is split to 0.01 $/h for CPU (50%), 0.002 $/h for

RAM (10%) and 0.008 for HDD (40%). We divide those prices by the respective component benchmark results and add them up: 0.01/140 + 0.002/427 + 0.008/32649 = 7.14E-05. We conduct that for every tariff. Comparing the resulting component weighed price (Table 6-5, last column), we realize that for this profile, B is now the preferred tariff in terms of price-performance ratio. This small example explains the calculation scheme. Please refer to the validation section for a comprehensive calculation.

Table 6-5. Numerical example for performance-ratio calculation (profile: 50%,10%,40%)

Provider tariff	Benchmark result $\overline{X}_i$ (Benchmark Points, BP)			Announced price ($/h)	Performance weighed component price ($/h/BP)			Composed Total Weighed Price ($/h/BP)
	CPU	RAM	HDD		CPU	RAM	HDD	
A	140	427	32,649	0.02	1.00E-02	2.00E-03	8.00E-03	7.14E-05
B	851	1,844	163,670	0.06	3.00E-02	6.00E-03	2.40E-02	3.53E-05
C	6,422	35,969	584,052	0.48	2.40E-01	4.80E-02	1.92E-01	3.74E-05

At this point we want to emphasize the key assumptions we need to make: (1) The applied benchmark is an appropriate performance metric for the component with respect to the application scenario. (2) The benchmark metric and the component's performance have a linear relation, e.g. double points in the benchmark mean double performance. (3) Performance differences perfectly correlate with willingness-to-pay differences, i.e. there is a linear relation between the performance the consumer receives and his/her willingness-to-pay.

6.4.4 Ranking and output

After we computed the price-performance ratio, we use it to rank the provider tariffs. Tariffs not matching the specified qualitative criteria have already been filtered out, so we can present this list to the user. If the consumer prefers an automatic selection (e.g. for immediate deployment), the first provider tariff on the list is the one with the lowest price per performance unit and hence would be chosen for the consumer's task.

6.5 Validation

In order to show the practical feasibility and utility of our suggested approach, we collected price and performance data of 14 IaaS tariffs. For the purpose of this proof-of-concept demonstration, we used only pay-per-use tariffs without upfront costs and one-time benchmark results. Based on this data, we first conduct an overall tariff performance comparison and subsequently, we use a consumer demand request to perform an application-specific tariff selection.

For our simulation, we use the data of three large cloud providers: Amazon, Azure and Rackspace. For legal purposes, we allocated aliases to the tariffs (Table 6-6). Providers are denominated by letters A, B and C while their tariffs are distinguished by numbers.

Table 6-6. Validation data set

Tariff Name	CPU	RAM	HDD	Price ($)	UnixBench Perf. Score
A1	1	0.613	10	$0.020	78.1
A2	2	3.75	410	$0.130	382.2
A3	6.5	17.1	420	$0.460	941.9
A4	26	68.4	1690	$1.840	2381.8
A5	1	0.613	10	$0.020	126.1
A6	2	3.75	410	$0.120	436.1
A7	6.5	17.1	420	$0.410	1075.3
A8	26	68.4	1690	$1.640	2513.2
B1	1	0.768	20	$0.060	933.1
B2	2	3.5	135	$0.120	1459.6
B3	8	14	605	$0.480	3088
C1	1	1	40	$0.060	329.8
C2	2	4	160	$0.240	592.7
C3	6	15	620	$0.900	1131.8

The structure for the analysis of an applied use case on the set of cloud providers is as follows: we first apply our model against the dataset to perform a general tariff comparison. Afterwards we conduct an application-specific tariff selection to show how a load profile is applied against data. In the application-specific version, we use a list of primary qualitative criteria to filter the tariffs for restrictions.

6.5.1 General tariff comparison

This analysis contains an unfiltered ranking of the actual price/performance ratio offered through the different cloud provider. Table 6-7 shows the prices as well as the UnixBench performance scores. After running the ranking mechanism against the dataset, we receive a result as shown by Table 6-7.

Table 6-7. Results of the general price-performance-comparison

Tariffs	Price ($/h)	UnixBench Performance Score	Price/Performance Score
B1	$0.06	933.1	6.43018E-05
B2	$0.12	1459.6	8.22143E-05
B3	$0.48	3088.0	0.00015544
A5	$0.02	126.7	0.000157853
C1	$0.06	329.8	0.000181928
A1	$0.02	78.1	0.000256082
A6	$0.12	436.1	0.000275166
A2	$0.13	382.2	0.000340136
A7	$0.41	1075.3	0.000381289
C2	$0.24	592.7	0.000404927
A3	$0.46	941.9	0.000488375
A8	$1.64	2513.2	0.000652555
A4	$1.84	2381.8	0.000772525
C3	$0.90	1131.8	0.000795193

Looking at the results, there are some interesting findings. As stated before, there are large differences between the provider prices and performance. Even one provider, e.g. provider A, which has data centers distributed across four continents world-wide, already shows heterogeneous results in terms of price-performance ratio. Overall, we see that the order of the price/performance score is different from the order by price or performance alone. Additionally, performance and price diverge among providers even in absolute terms, giving less performance at a higher price (e.g. B2 and A7). Provider B delivers the best price/performance score across all instances. An interesting fact is that the micro instance with the lowest resource component set has the best price/performance score. Compared to the micro instances of provider A for example it delivers a nine times better performance while the price per hour is only three times higher. Apart from that, the results indicate that the larger instances have a worse price/performance score in general. This observation is in accordance with findings in literature (e.g. Lenk et al. 2011; Jiang 2012).

6.5.2 Application-specific comparison

For the profiled version of our model we now apply consumer-provided constraints. First of all, we define three different primary qualitative criteria to filter for: Number of CPUs [4-∞]; GB RAM [14-∞]; GB HDD [100; ∞]. As introduced in the use case scenario, the profiled version of our model allows customers to assign each component with an importance weight. We assume an in-memory database use case, hence the primary focus component is RAM. Because of this, the derived user load vector for (CPU, RAM, HDD) is: $\vec{l} = (20\%, 70\%, 10\%)$. Table 6-8 shows the results.

Table 6-8. Profiled benchmark results

Provider tariff	Benchmark result $\overline{X}_i$ (Benchmark Points, BP)			Announced price ($/h)	Performance weighed component price ($/h/BP)			Composed Total Weighed Price ($/h/BP)
	CPU	RAM	HDD		CPU	RAM	HDD	
B3	6422	35969	584052	0.48	1.49E-05	9.34E-06	8.22E-08	2.44E-05
A7	2489	5429	615451	0.41	3.29E-05	5.29E-05	6.66E-08	8.59E-05
A3	2144	4585	522814	0.46	4.29E-05	7.02E-05	8.80E-08	1.13E-04
A8	7545	14534	4457800	1.64	4.35E-05	7.90E-05	3.68E-08	1.22E-04
C3	3512	7118	183571	0.90	5.13E-05	8.85E-05	4.90E-07	1.40E-04
A4	6849	13014	449711	1.84	5.37E-05	9.90E-05	4.09E-07	1.53E-04
Other tariffs ignored due to qualitative constraints								

First, we see that due to the filter constraints, many tariffs will not be calculated as they do not meet the requirements. The best suited tariff suggested is B3. Due to the emphasis on RAM performance, these results show a slightly different ranking compared to the general system comparison (C3 and A4 are exchanged). Due to the limited number of tariffs, the differences between tariffs are quite high and thus the per-

formance-adjusted rating is quite stable. In a real scenario, with a high number of tariffs, the selection support based on application-specific computation demands becomes increasingly important to identify the right tariff for a given demand. Using this proof-of-concept, we illustrated how our approach works on real data and that it helps to increase performance transparency for an application demand profile specified by the consumer.

6.6 Conclusion

Cloud Computing turns into a commodity. This requires efficient sourcing solutions, especially as market size and complexity increases. In this paper, we suggested a first approach for a broker model to increase price transparency with respect to performance. By constantly monitoring provider performance, we create a knowledge base about the strength and weaknesses of the provider market down to a component level. We use this knowledge to find the best offer in terms of price-performance ratio for a given user computation scenario. Due to the formalization of the model, it can be implemented into a decision support system to provide automated assistance when sourcing IaaS resources. As such, it is of practical use to businesses who want to outsource their computing needs into the cloud. From a research perspective, the data collected by the broker over time provides deep insights into the structure of the IaaS market and an implementation could be used to experiment with different brokerage approaches.

However, there are some limitations to our work. First, the approach assumes that the used benchmarks are representative for the performance experienced by the application. While this should often be the case, there might be applications, which cannot adequately be captured by the benchmarks and the load profile specified by the user. In addition, we only focused on finding the right tariff for a single VM instance, while more complex setups might need a combined approach optimizing different application profiles at once. Second, there is the assumption that there is a linear relationship between benchmark value and performance value (with respect to the value it delivers through the application). And third, the consumer still has to estimate some crucial parameters which might be difficult if the experience is missing on his side. Especially, specifying an appropriate task load profile for the application-specific tariff comparison is not straight-forward. We suggested a first theoretical approach using empirical data to estimate an applicable profile. However, the effectiveness of this approach still has to be verified in more detail.

For future work, we plan a thorough refinement of the model to include more component benchmarks (e.g. Network speed) and further unify the diversity of pricing options that providers expose (e.g. charges of I/O operations). In our vision of a future IaaS broker, the consumer selects one benchmark best matching his computing purpose from a catalogue of application benchmarks (e.g. cf. Phoronix Media 2013) to find his preferred provider. After deployment, the broker tracks the user's load profile and use it for later sourcing decisions to get a very precise idea of the user's computing needs. This could lead to a continuous improvement relationship, where the broker keeps on optimizing the supply for the consumer's demand handling price changes, different subscription models or payment cycles autonomously and regarding dependencies among components in a more complex setup (e.g. common storage infrastructure). At that point, sourcing Cloud Computing resources should be as easy as turning on a faucet.

References

Adomavicius, G, and A Tuzhilin. 2005. “Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions.” *Knowledge and Data Engineering, IEEE Transactions on* 17 (6): 734–49.

Agarwal, Ritu, and Vasant Dhar. 2014. “Editorial — Big Data, Data Science, and Analytics: The Opportunity and Challenge for IS Research.” *Information Systems Research* 25 (3): 443–48. doi:10.1287/isre.2014.0546.

Ahearne, Michael, C B Bhattacharya, and Thomas Gruen. 2005. “Antecedents and Consequences of Customer-Company Identification: Expanding the Role of Relationship Marketing.” *The Journal of Applied Psychology* 90 (3): 574–85. doi:10.1037/0021-9010.90.3.574.

Akerlof, George a. 1970. “The Market for ‘Lemons’: Quality Uncertainty and the Market Mechanism.” *The Quarterly Journal of Economics* 84 (3): 488–500. doi:10.2307/1879431.

Andrecut, M. 2013. “Portfolio Optimization in R.” *arXiv* 1 (1307.0450v1).

Anthony, R N. 1965. “Planning and Control Systems: A Framework for Analysis.” *Studies in Management Control*. Harvard University Graduate School of Business Administration, Cambridge, MA.

Antweiler, Werner, and Murray Z. Frank. 2004. “Is All That Talk Just Noise? The Information Content of Internet Stock Message Boards.” *The Journal of Finance* 59 (3): 1259–94. doi:10.1111/j.1540-6261.2004.00662.x.

Aral, Sinan, and Dylan Walker. 2012. “Identifying Influential and Susceptible Members of Social Networks.” *Science (New York, N.Y.)* 337 (6092): 337–41. doi:10.1126/science.1215842.

Armbrust, Michael, Anthony D Joseph, Randy H Katz, David A Patterson, Armando Fox, Rean Griffith, Anthony D Joseph, et al. 2009. *Above the Clouds: A Berkeley View of Cloud Computing*. UCB/EECS-2009-28. Technical Report. Berkeley.

Arrow, Kenneth J, Robert Forsythe, Michael Gorham, Robert Hahn, Robin Hanson, John O Ledyard, Saul Levmore, et al. 2008. “The Promise of Prediction Markets.” *Science (New York, N.Y.)* 320 (5878): 877–78. doi:10.1126/science.1157679.

Avery, Christopher, Judith Chevalier, and Richard Zeckhauser. 2009. *The “ CAPS ” Prediction System and Stock Market Returns*. No. RWP09-011. HKS Working

Paper.

———. 2011. *The "CAPS" Prediction System and Stock Market Returns*. RWP11-028. Faculty Research Working Paper Series. Cambridge.

Baba, N., and N. Inoue. 2002. "Utilization of Soft Computing Techniques for Constructing Reliable Decision Support Systems for Dealing Stocks." In *Proceedings of the 2002 International Joint Conference on Neural Networks. IJCNN'02 (Cat. No.02CH37290)*, 2150–55. IEEE. doi:10.1109/IJCNN.2002.1007474.

Beane, T.P., and D.M. Ennis. 1987. "Market Segmentation: A Review." *European Journal of Marketing* 21 (5): 20–42. doi:10.1108/EUM0000000004695.

Benner, Mary J, and Michael L Tushman. 2003. "Exploitation, Exploration, and Process Management: The Productivity Dilemma Revisited." *The Academy of Management Review* 28 (2): 238. doi:10.2307/30040711.

Beraldi, P, A Violi, and F De Simone. 2011. "A Decision Support System for Strategic Asset Allocation." *Decision Support Systems* 51 (3). Elsevier B.V.: 549–61. doi:10.1016/j.dss.2011.02.017.

Berg, Joyce E., and Thomas A. Rietz. 2003. "Prediction Markets as Decision Support Systems." *Information Systems Frontiers* 5 (1): 79–93. doi:10.1023/A:1022002107255.

Berlecon Research GmbH. 2010. "Das Wirtschaftliche Potenzial Des Internet Der Dienste." *http://www.berlecon.de/studien/downloads/Berlecon_IDD_ExecSum.pdf; Accessed 2013-04-23*. http://www.berlecon.de/studien/downloads/Berlecon_IDD_ExecSum.pdf.

Binnig, Carsten, Donald Kossmann, Tim Kraska, and Simon Loesing. 2009. "How Is the Weather Tomorrow?" In *Proceedings of the Second International Workshop on Testing Database Systems - DBTest '09*, 1. New York, New York, USA: ACM Press.

Bobadilla, Jesús, Fernando Ortega, Antonio Hernando, and Abraham Gutiérrez. 2013. "Recommender Systems Survey." *Knowledge-Based Systems*.

Bogers, Toine, and Antal van den Bosch. 2011. "Fusing Recommendations for Social Bookmarking Web Sites." *International Journal of Electronic Commerce* 15 (3): 31–72. doi:10.2753/JEC1086-4415150303.

Bollen, Johan, Huina Mao, and Xiao-jun Zeng. 2011. "Twitter Mood Predicts the Stock

Market." *Journal of Computational Science* 2 (1): 1–8.

Bortz, Jürgen. 2005. *Statistik Für Human- Und Sozialwissenschaftler (in German)*. 6th ed. Heidelberg: Springer Berlin Heidelberg.

Buetow, Gerald W, Ronald Sellers, Donald Trotter, Elaine Hunt, and Willie A Whipple. 2002. "The Benefits of Rebalancing." *The Journal of Portfolio Management* 28 (2): 23–32. doi:10.3905/jpm.2002.319829.

Burgelman, Robert A. 2002. "Strategy as Vector and the Inertia of Coevolutionary Lock-In." *Administrative Science Quarterly* 47 (2): 325. doi:10.2307/3094808.

Buyya, Rajkumar, Chee Shin Yeo, Srikumar Venugopal, James Broberg, and Ivona Brandic. 2009. "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility." *Future Generation Computer Systems* 25 (6). Elsevier B.V.: 599–616. doi:10.1016/j.future.2008.12.001.

Campbell, John Y, Martin Lettau, Burton G Malkiel, and Yexiao Xu. 2001. "Have Individual Stocks Become More Volatile? An Empirical Exploration of Idiosyncratic Risk." *The Journal of Finance* 56 (1): 1–43. doi:10.1111/0022-1082.00318.

Carhart, MARK M. 1997. "On Persistence in Mutual Fund Performance." *The Journal of Finance* 52 (1): 57–82. doi:10.1111/j.1540-6261.1997.tb03808.x.

Chan, Yolande E, and Blaize Horner Reich. 2007. "IT Alignment: What Have We Learned?" *Journal of Information Technology* 22 (4): 297–315. doi:10.1057/palgrave.jit.2000109.

Chang, Ray M., Robert J. Kauffman, and Youngok Kwon. 2014. "Understanding the Paradigm Shift to Computational Social Science in the Presence of Big Data." *Decision Support Systems* 63. Elsevier B.V.: 67–80. doi:10.1016/j.dss.2013.08.008.

Chard, Kyle, Kris Bubendorfer, and Peter Komisarczuk. 2010. "High Occupancy Resource Allocation for Grid and Cloud Systems, a Study with DRIVE." In , 73–84. HPDC '10. New York, NY, USA: ACM.

Chen, Hsinchun, Roger H. L. Chiang, and Veda C Storey. 2012. "Business Intelligence and Analytics: From Big Data To Big Impact." *MIS Quarterly* 36 (4): 1165–88. doi:10.1145/2463676.2463712.

Chen, Li, Paulo Goes, James R. Marsden, and Zhongju Zhang. 2009. "Design and Use of Preference Markets for Evaluation of Early Stage Technologies." *Journal of Management Information Systems* 26 (3): 45–70. doi:10.2753/MIS0742-

1222260302.

Cho, Vincent. 2010. “MISMIS – A Comprehensive Decision Support System for Stock Market Investment.” *Knowledge-Based Systems* 23 (6). Elsevier B.V.: 626–33. doi:10.1016/j.knosys.2010.04.009.

Chopra, Vijay Kumar, and William T Ziemba. 1993. “The Effect of Errors in Means, Variances, and Covariances on Optimal Portfolio Choice.” *The Journal of Portfolio Management* 19 (2): 6–11. doi:10.3905/jpm.1993.409440.

Chou, SCT, CC Yang, Chi-huang Chen, and Feipei Lai. 1996. “A Rule-Based Neural Stock Trading Decision Support System.” In *Computational Intelligence for Financial Engineering, 1996., Proceedings of the IEEE/IAFE 1996 Conference on*, 148–54.

Claus, Barbara. 2013. *Morningstar Research Report Allianz Thesaurus AT EUR (September 20, 2012)*.

Cloudspectator. 2012. “Lunacloud, Amazon, and Rackspace Benchmark Report.” *Http://www.cloudspectator.com/services/reports/performance/lunacloud-Amazon-and-Rackspace-Performance-Report/*.

Cohen, KJ, and JA Pogue. 1967. “An Empirical Evaluation of Alternative Portfolio-Selection Models.” *The Journal of Business* 40.

comdirect Bank AG. 2013. “Preis- Und Leistungsverzeichnis.” https://www.comdirect.de/cms/media/corp0099_preisleistungsverzeichnis.pdf?eCRMLink=S#preis.

Cooper, Brian, Adam Silberstein, Erwin Tam, Raghu Ramakrishnan, and Russell Sears. 2010. *Benchmarking Cloud Serving Systems with YCSB*.

Cortizo, José Carlos, Francisco M. Carrero, and José María Gómez. 2011. “Introduction to the Special Issue: Mining Social Media.” *International Journal of Electronic Commerce* 15 (3): 5–8. doi:10.2753/JEC1086-4415150301.

Dahan, Ely, Arina Soukhoroukova, and Martin Spann. 2010. “New Product Development 2.0: Preference Markets-How Scalable Securities Markets Identify Winning Product Concepts and Attributes.” *Journal of Product Innovation Management* 27 (7): 937–54. doi:10.1111/j.1540-5885.2010.00763.x.

Davenport, Thomas H. 2006. “Competing On Analytics.” *Harvard Business Review* 84 (1): 98–107.

DeMiguel, V., L. Garlappi, and R. Uppal. 2007. “Optimal Versus Naive Diversification: How Inefficient Is the 1/N Portfolio Strategy?” *Review of Financial Studies* 22

(5): 1915–53. doi:10.1093/rfs/hhm075.

Dewally, Michaël, and Louis Ederington. 2006. “Reputation, Certification, Warranties, and Information as Remedies for Seller-Buyer Information Asymmetries: Lessons from the Online Comic Book Market.” *The Journal of Business* 79 (2): 693–729. doi:10.1086/499169.

DiNucci, Darcy. 1999. “Fragmented Future.” *Print* 53 (4): 32, 221–22.

Domian, Dale L, David A Louton, and Marie D. Racine. 2007. “Diversification in Portfolios of Individual Stocks: 100 Stocks Are Not Enough.” *The Financial Review* 42 (4): 557–70. doi:10.1111/j.1540-6288.2007.00183.x.

Dong, Jichang, Helen S. Du, Shouyang Wang, Kang Chen, and Xiaotie Deng. 2004. “A Framework of Web-Based Decision Support Systems for Portfolio Selection with OLAP and PVM.” *Decision Support Systems* 37 (3): 367–76. doi:10.1016/S0167-9236(03)00034-4.

El Kihal, Siham, Christian Schlereth, and Bernd Skiera. 2012. “Price Comparison for Infrastructure-as-a-Service.” In *ECIS 2012 Proceedings*, 1–12. Paper 161.

Elton, Edwin J, and Martin J Gruber. 1997. “Modern Portfolio Theory, 1950 to Date.” *Journal of Banking & Finance* 21 (11-12): 1743–59. doi:10.1016/S0378-4266(97)00048-4.

Evans, John L., and Stephen H. Archer. 1968. “Diversification and the Reduction of Dispersion: An Empirical Analysis.” *The Journal of Finance* 23 (5): 761–67. doi:10.1111/j.1540-6261.1968.tb00315.x.

Facebook.com. 2014. “Accessing Your Facebook Info | Facebook Help Center.” https://www.facebook.com/help/405183566203254/.

Fama, Eugene F., and Kenneth R. French. 1993. “Common Risk Factors in the Returns on Stocks and Bonds.” *Journal of Financial Economics* 33 (1): 3–56. doi:10.1016/0304-405X(93)90023-5.

Fennell, Geraldine, Greg M. Allenby, Sha Yang, and Yancy Edwards. 2003. “The Effectiveness of Demographic and Psychographic Variables for Explaining Brand and Product Category Use.” *Quantitative Marketing and Economics* 1 (2): 223–44. doi:10.1023/A:1024686630821.

Forsythe, Robert, Forrest Nelson, GR Neumann, and Jack Wright. 1992. “Anatomy of an Experimental Political Stock Market.” *The American Economic Review* 82 (5): 1142–61.

Frischmann, Tanja, Oliver Hinz, and Bernd Skiera. 2012. “Retailers’ Use of Shipping

Cost Strategies: Free Shipping or Partitioned Prices?" *International Journal of Electronic Commerce* 16 (3): 65–88. doi:10.2753/JEC1086-4415160303.

Garg, Saurabh Kumar, Steve Versteeg, and Rajkumar Buyya. 2013. "A Framework for Ranking of Cloud Computing Services." *Future Generation Computer Systems* 29 (4). Elsevier B.V.: 1012–23. doi:10.1016/j.future.2012.06.006.

Gartner. 2016. "Gartner Says Worldwide Public Cloud Services Market Is Forecast to Reach $204 Billion in 2016." http://www.gartner.com/newsroom/id/3188817.

Ghysels, Eric, Pedro Santa-Clara, and Rossen Valkanov. 2005. "There Is a Risk-Return Trade-off after All." *Journal of Financial Economics* 76 (3): 509–48. doi:10.1016/j.jfineco.2004.03.008.

Giles, Jim. 2005. "Internet Encyclopaedias Go Head to Head." *Nature* 438 (15): 900–901. doi:10.1038/438900a.

Goes, Paulo B. 2014. "Big Data and IS Research – Editor's Comments." *MIS Quarterly* 38 (3): iii – viii.

Google. 2013. "Google Compute Engine Pricing - Cloud Platform." *Https://cloud.google.com/pricing/compute-Engine*. https://cloud.google.com/pricing/compute-engine.

Gorry, George Anthony, and Michael S Scott Morton. 1971. "A Framework for Management Information Systems." *Sloan Management Review* 13 (1): 50–70.

Gottschlich, Jörg, and Oliver Hinz. 2014. "A Decision Support System for Stock Investment Recommendations Using Collective Wisdom." *Decision Support Systems* 59 (March): 52–62. doi:10.1016/j.dss.2013.10.005.

Groth, Sven S, and Jan Muntermann. 2009. "Supporting Investment Management Processes with Machine Learning Techniques." In *Wirtschaftsinformatik Proceedings*. Paper 107.

Guo, H U I, and Robert F Whitelaw. 2006. "Uncovering the Risk – Return Relation in the Stock Market" LXI (3).

Gupta, Anil K., Ken G. Smith, and Christina E. Shalley. 2006. "The Interplay Between Exploration and Exploitation." *Academy of Management Journal* 49 (4): 693–706. doi:10.5465/AMJ.2006.22083026.

Hausman, J. A. 1978. "Specification Tests in Econometrics." *Econometrica* 46 (6): 1251. doi:10.2307/1913827.

He, Jianming, and Wesley W Chu. 2010. *A Social Network-Based Recommender*

System (SNRS). Springer.

He, Zi-Lin, and Poh-Kam Wong. 2004. “Exploration vs. Exploitation: An Empirical Test of the Ambidexterity Hypothesis.” *Organization Science* 15 (4): 481–94. doi:10.1287/orsc.1040.0078.

Hendershott, Terrence, CHARLES M. Jones, and ALBERT J. Menkveld. 2011. “Does Algorithmic Trading Improve Liquidity?” *The Journal of Finance* 66 (1): 1–33. doi:10.1111/j.1540-6261.2010.01624.x.

Henderson, JC, and N Venkatraman. 1993. “Strategic Alignment: Leveraging Information Technology for Transforming Organizations.” *IBM Systems Journal* 32 (1): 472–84.

Hevner, Alan R., Salvatore T. March, Jinsoo Park, and Sudha Ram. 2004. “Design Science in Information Systems Research.” *Mis Quarterly* 28 (1): 75–105.

Hill, Shawndra, and Noah Ready-Campbell. 2011. “Expert Stock Picker: The Wisdom of (Experts In) Crowds.” *International Journal of Electronic Commerce* 15 (3): 73–102. doi:10.2753/JEC1086-4415150304.

Hinz, Oliver, and Jochen Eckert. 2010. “The Impact of Search and Recommendation Systems on Sales in Electronic Commerce.” *Business & Information Systems Engineering* 2 (2): 67–77. doi:10.1007/s12599-010-0092-x.

Hinz, Oliver, IH Hann, and Martin Spann. 2011. “Price Discrimination in E-Commerce? An Examination of Dynamic Pricing in Name-Your-Own Price Markets.” *MIS Quarterly* 35 (1): 81–98.

Hinz, Oliver, and Martin Spann. 2010. “Managing Information Diffusion in Name-Your-Own-Price Auctions.” *Decision Support Systems* 49 (4). Elsevier B.V.: 474–85. doi:10.1016/j.dss.2010.05.008.

Hitt, Michael A, Baibaia W Keats, and S. M. DeMarie. 1998. “Navigating in the New Competitive Landscape: Building Strategic Flexibility and Competitive Advantage in the 21st Century.” *Academy of Management Perspectives* 12 (4): 22–42. doi:10.5465/AME.1998.1333922.

Hosack, Bryan, Dianne Hall, David Paradice, and James F Courtney. 2012. “A Look Toward the Future: Decision Support Systems Research Is Alive and Well.” *Journal of the Association for Information Systems* 13 (5): 315–40.

Huang, Zan, Wingyan Chung, and Hsinchun Chen. 2004. “A Graph Model for E-Commerce Recommender Systems.” *Journal of the American Society for Information Science and Technology* 55 (3): 259–74.

Hull, John C. 2007. *Risk Management and Financial Institutions*. 3rd ed. New Jersey: Pearson Prentice Hall.

Iosup, A, S Ostermann, M N Yigitbasi, R Prodan, T Fahringer, and D H J Epema. 2011. "Performance Analysis of Cloud Computing Services for Many-Tasks Scientific Computing." *IEEE Transactions on Parallel and Distributed Systems* 22 (6): 931–45. doi:10.1109/TPDS.2011.66.

Jannach, D, M Zanker, A Felfernig, and G Friedrich. 2010. *Recommender Systems: An Introduction*. Cambridge University Press.

Jiang, Qingye. 2012. "Virtual Machine Performance Comparison of Public IaaS Providers in China." In *2012 IEEE Asia Pacific Cloud Computing Congress (APCloudCC)*, 16–19. Ieee. doi:10.1109/APCloudCC.2012.6486504.

Kass, Robert, and Tim Finin. 1988. "Modeling the User in Natural Language Systems." *Computational Linguistics* 14 (3): 5–22.

Kim, Changsu, Jaeyong Song, and Atul Nerkar. 2012. "Learning and Innovation: Exploitation and Exploration Trade-Offs." *Journal of Business Research* 65 (8). Elsevier Inc.: 1189–94. doi:10.1016/j.jbusres.2011.07.006.

Kim, Heung-Nam, Ae-Ttie Ji, Inay Ha, and Geun-Sik Jo. 2010. "Collaborative Filtering Based on Collaborative Tagging for Enhancing the Quality of Recommendation." *Electronic Commerce Research and Applications* 9 (1): 73–83.

Kissel, Robert. 2013. *Science of Algorithmic Trading and Portfolio Management*. Academic Press.

Konno, Hiroshi, and Rei Yamamoto. 2003. "Minimal Concave Cost Rebalance of a Portfolio to the Efficient Frontier." *Mathematical Programming* 97 (3): 571–85. doi:10.1007/s10107-003-0428-0.

Krokhmal, Pavlo, Jonas Palmquist, and Stanislav Uryasev. 2002. "PORTFOLIO OPTIMIZATION WITH CONDITIONAL VALUE-AT-RISK OBJECTIVE AND CONSTRAINTS." *Journal of Risk* 4 (2): 11–27.

Krumm, John, Nigel Davies, and Chandra Narayanaswami. 2008. "User-Generated Content." *IEEE Pervasive Computing* 7 (4): 10–11. doi:10.1109/MPRV.2008.85.

Kuo, RJ, CH Chen, and YC Hwang. 2001. "An Intelligent Stock Trading Decision Support System through Integration of Genetic Algorithm Based Fuzzy Neural Network and Artificial Neural Network." *Fuzzy Sets and Systems* 118: 21–45.

Larrick, Richard P, and Jack B. Soll. 2006. "Intuitions About Combining Opinions: Misappreciation of the Averaging Principle." *Management Science* 52 (1): 111–

27. doi:10.1287/mnsc.1050.0459.

Leimeister, Jan Marco. 2010. “Collective Intelligence.” *Business & Information Systems Engineering* 2 (4): 245–48. doi:10.1007/s12599-010-0114-8.

Lenk, Alexander, Michael Menzel, Johannes Lipsky, Stefan Tai, and Philipp Offermann. 2011. “What Are You Paying For? Performance Benchmarking for Infrastructure-as-a-Service Offerings.” In *2011 IEEE 4th International Conference on Cloud Computing*, 484–91. IEEE. doi:10.1109/CLOUD.2011.80.

Levinthal, Daniel A., and James G. March. 1993. “The Myopia of Learning.” *Strategic Management Journal* 14 (S2): 95–112. doi:10.1002/smj.4250141009.

Li, Ang, Xiaowei Yang, Srikanth Kandula, and Ming Zhang. 2010. “CloudCmp: Comparing Public Cloud Providers.” In *Proceedings of the 10th Annual Conference on Internet Measurement - IMC ’10*, 1–14. New York, New York, USA: ACM Press.

Li, Qing, Jia Wang, Yuanzhu Peter Chen, and Zhangxi Lin. 2010. “User Comments for News Recommendation in Forum-Based Social Media.” *Information Sciences* 180 (24). Elsevier Inc.: 4929–39. doi:10.1016/j.ins.2010.08.044.

Li, Yingjie, Shushang Zhu, Donghui Li, and Duan Li. 2013. “Active Allocation of Systematic Risk and Control of Risk Sensitivity in Portfolio Optimization.” *European Journal of Operational Research* 228 (3): 556–70. doi:10.1016/j.ejor.2013.02.016.

Li, Yung-Ming, Chun-Te Wu, and Cheng-Yang Lai. 2013. “A Social Recommender Mechanism for E-Commerce: Combining Similarity, Trust, and Relationship.” *Decision Support Systems* 55 (3): 740–52. doi:http://dx.doi.org/10.1016/j.dss.2013.02.009.

Linsmeier, Thomas J., and Neil D. Pearson. 2000. “Value at Risk.” *Financial Analysts Journal* 56 (2): 47–67. doi:10.2469/faj.v56.n2.2343.

Liu, NK, and KK Lee. 1997. “An Intelligent Business Advisor System for Stock Investment.” *Expert Systems* 14 (3).

Lorge, I, D Fox, J Davitz, and M Brenner. 1958. “A Survey of Studies Contrasting the Quality of Group Performance and Individual Performance, 1920-1957.” *Psychological Bulletin* 59 (6): 257–72.

Lucey, Brian M., and Michael Dowling. 2005. “The Role of Feelings in Investor Decision-Making.” *Journal of Economic Surveys* 19 (2): 211–37. doi:10.1111/j.0950-0804.2005.00245.x.

Lukyanenko, Roman, Jeffrey Parsons, Yolanda F Wiersma, Roman Lukyanenko, and Jeffrey Parsons. 2014. “The IQ of the Crowd : Understanding and Improving Information Quality in Structured User-Generated Content The IQ of the Crowd : Understanding and Improving Information Quality in Structured User-Generated Content,” no. January 2015.

Mann, H. B., and D. R. Whitney. 1947. “On a Test of Whether One of Two Random Variables Is Stochastically Larger than the Other.” *The Annals of Mathematical Statistics* 18 (1): 50–60. doi:10.1214/aoms/1177730491.

March, James G. 1991. “Exploration and Exploitation in Organizational Learning.” Edited by William H Starbuck and Peter S Whalen. *Organization Science* 2 (1). JSTOR: 71–87. doi:10.1287/orsc.2.1.71.

Markowitz, Harry M. 1952. “Portfolio Selection.” *The Journal of Finance* 7 (1): 77–91. doi:10.1111/j.1540-6261.1952.tb01525.x.

———. 1991. “Foundations of Portfolio Theory.” *The Journal of Finance* 46 (2): 469–77. doi:10.1111/j.1540-6261.1991.tb02669.x.

Mattern, Friedemann, and Christian Flörkemeier. 2010. “Vom Internet Der Computer Zum Internet Der Dinge.” *Informatik-Spektrum* 33 (2): 107–21. doi:10.1007/s00287-010-0417-7.

McAlexander, JH, JW Schouten, and HF Koenig. 2002. “Building Brand Community.” *The Journal of Marketing* 66 (1): 38–54.

McGhee, India, Jennifer Bayzick, April Kontostathis, Lynne Edwards, Alexandra McBride, and Emma Jakubowski. 2011. “Learning to Identify Internet Sexual Predation.” *International Journal of Electronic Commerce* 15 (3): 103–22. doi:10.2753/JEC1086-4415150305.

Mell, P M, and T Grance. 2011. *The NIST Definition of Cloud Computing*. Gaithersburg, MD. doi:10.6028/NIST.SP.800-145.

Melville, Nigel, Kenneth Kraemer, and Vijay Gurbaxani. 2004. “Review: Information Technology and Organizational Performance: An Integrative Model of IT Business Value.” *MISQ* 28 (2). Minneapolis, MN, USA: Society for Information Management and The Management Information Systems Research Center: 283–322.

Mintzberg, Henry. 1971. “Managerial Work: Analysis from Observation.” *Management Science* 18 (2): B97–110. doi:10.2307/2629532.

Mittermayer, Marc-André. 2004. “Forecasting Intraday Stock Price Trends with Text

Mining Techniques." In *37th Annual Hawaii International Conference on System Sciences, 2004. Proceedings of the*, 00:10 pp. IEEE. doi:10.1109/HICSS.2004.1265201.

Montaner, Miquel, B López, and JL De La Rosa. 2003. "A Taxonomy of Recommender Agents on the Internet." *Artificial Intelligence Review* 19: 285–330.

Moynihan, Gary P., Prasad Purushothaman, Robert W. McLeod, and William G. Nichols. 2002. "DSSALM: A Decision Support System for Asset and Liability Management." *Decision Support Systems* 33 (1): 23–38. doi:10.1016/S0167-9236(01)00131-2.

Muntermann, Jan. 2009. "Towards Ubiquitous Information Supply for Individual Investors: A Decision Support System Design." *Decision Support Systems* 47 (2). Elsevier B.V.: 82–92. doi:10.1016/j.dss.2009.01.003.

Narang, Rishi K. 2009. *Inside the Black Box: The Simple Truth about Quantitative Trading*. New Jersey: John Wiley & Sons.

Nelson, P. 1970. "Information and Consumer Behavior." *The Journal of Political Economy* 78 (2): 311–29.

Nofer, Michael, and Oliver Hinz. 2014. "Are Crowds on the Internet Wiser than Experts? The Case of a Stock Prediction Community." *Journal of Business Economics* 84 (3): 303–38. doi:10.1007/s11573-014-0720-x.

Nöth, Simon. 2013. *Morningstar Research Report DWS Deutschland (July 19, 2013)*.

Ogryczak, Włodzimierz, and Tomasz Sliwinski. 2010. "Efficient Portfolio Optimization with Conditional Value at Risk." In *Computer Science and Information Technology (IMCSIT), Proceedings of the 2010 International Multiconference on*, 901–8.

Page, Rich. 2008. "Web Metrics 101 – What DO All These Terms Mean?" *Blog*. http://www.makeuseof.com/tag/web-metrics-101-what-do-all-these-terms-mean/.

Palmer, Jonathan W. 2002. "Web Site Usability, Design, and Performance Metrics." *Information Systems Research* 13 (2): 151–67. doi:10.1287/isre.13.2.151.88.

Patiniotakis, Ioannis, Yiannis Verginadis, and Gregoris Mentzas. 2014. "Preference-Based Cloud Service Recommendation as a Brokerage Service." In *Proceedings of the 2nd International Workshop on CrossCloud Systems - CCB '14*, 1–6. Paper 5. doi:10.1145/2676662.2676677.

Patterson, Scott. 2012. "Breakdown: A Glimpse Inside the 'Flash Crash.'" *Wall Street*

Journal Online. http://online.wsj.com/news/articles/SB10001424052702303296604577454330066039896.

Pereira, RE. 2000. "Optimizing Human-Computer Interaction for the Electronic Commerce Environment." *Journal of Electronic Commerce Research* 1 (1): 23–44.

Phoronix Media. 2013. "Openbenchmarking.org." *Http://openbenchmarking.org/tests/pts; Accessed 2013-05-02*. http://openbenchmarking.org/tests/pts.

Plaxco, Lisa M, and Robert D Arnott. 2002. "Rebalancing a Global Policy Benchmark." *The Journal of Portfolio Management* 28 (2): 9–22. doi:10.3905/jpm.2002.319828.

Rashid, Al Mamunur, George Karypis, and John Riedl. 2008. "Learning Preferences of New Users in Recommender Systems: An Information Theoretic Approach." *ACM SIGKDD Explorations Newsletter* 10 (2): 90–100.

Repschlaeger, Jonas, S Wind, R Zarnekow, and Klaus Turowski. 2013. "Decision Model for Selecting a Cloud Provider: A Study of Service Model Decision Priorities." In *AMCIS 2013 Proceedings*, 1–11, Paper 27.

Ries, Eric. 2011. *The Lean Startup: How Today's Entrepreneurs Use Continuous Innovation to Create Radically Successful Businesses*. Crown Books.

Riggins, Frederick J., and Samuel Fosso Wamba. 2015. "Research Directions on the Adoption, Usage, and Impact of the Internet of Things through the Use of Big Data Analytics." In *Proceedings of the Annual Hawaii International Conference on System Sciences*, 2015-March:1531–40. doi:10.1109/HICSS.2015.186.

Rodríguez, Rosa M, Macarena Espinilla, Pedro J Sánchez, and Luis Martínez-López. 2010. "Using Linguistic Incomplete Preference Relations to Cold Start Recommendations." *Internet Research* 20 (3): 296–315.

Ruiz, Eduardo J., Vagelis Hristidis, Carlos Castillo, Aristides Gionis, and Alejandro Jaimes. 2012. "Correlating Financial Time Series with Micro-Blogging Activity." *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining - WSDM '12*. New York, New York, USA: ACM Press, 513. doi:10.1145/2124295.2124358.

Schafer, JB, JA Konstan, and J Riedl. 2001. "E-Commerce Recommendation Applications." *Data Mining and Knowledge Discovery* 5: 115–53.

Schoder, Detlef, Johannes Putzke, Panagiotis Takis Metaxas, Peter A. Gloor, and Kai Fischbach. 2014. “Information Systems for ‘Wicked Problems.’” *Business & Information Systems Engineering* 6 (1): 3–10. doi:10.1007/s12599-013-0303-3.

Schumaker, Robert P., and Hsinchun Chen. 2009. “Textual Analysis of Stock Market Prediction Using Breaking Financial News.” *ACM Transactions on Information Systems* 27 (2): 1–19. doi:10.1145/1462198.1462204.

Sharpe, William F. 1992a. “Asset Allocation: Management Style and Performance Measurement.” *The Journal of Portfolio Management* 18 (2). Institutional Investor Journals: 7–19.

———. 1992b. “Asset Allocation: Management Style and Performance Measurement.” *The Journal of Portfolio Management* 18 (2): 7–19.

Sharpe, William F. 1966. “Mutual Fund Performance.” *The Journal of Business* 39 (1): 119–38. doi:10.1086/294846.

Shawky, Hany a., and David M. Smith. 2005. “Optimal Number of Stock Holdings in Mutual Fund Portfolios Based on Market Performance.” *The Financial Review* 40 (4): 481–95. doi:10.1111/j.1540-6288.2005.00120.x.

Shim, J.P. P, Merrill Warkentin, James F. Courtney, Daniel J. Power, Ramesh Sharda, and Christer Carlsson. 2002. “Past, Present, and Future of Decision Support Technology.” *Decision Support Systems* 33 (2): 111–26. doi:10.1016/S0167-9236(01)00139-7.

Simmons, Joseph P., Leif D. Nelson, Jeffrey Galak, and Shane Frederick. 2011. “Intuitive Biases in Choice versus Estimation: Implications for the Wisdom of Crowds.” *Journal of Consumer Research* 38 (1): 1–15. doi:10.1086/658070.

Simon, Herbert A. 1960. *The New Science of Management Decision*. New York: Harper & Brothers. doi:10.1037/13978-000.

Solis, Brian. 2016. “The Conversation Prism.” https://conversationprism.com/.

Soll, Jack B, and Richard P Larrick. 2009. “Strategies for Revising Judgment: How (and How Well) People Use Others’ Opinions.” *Journal of Experimental Psychology. Learning, Memory, and Cognition* 35 (3): 780–805. doi:10.1037/a0015145.

Spann, Martin, and Bernd Skiera. 2003. “Internet-Based Virtual Stock Markets for Business Forecasting.” *Management Science* 49 (10): 1310–26. doi:10.1287/mnsc.49.10.1310.17314.

Spiekermann, Sarah, Jens Grossklags, and Bettina Berendt. 2001. “E-Privacy in 2nd

Generation E-Commerce." In *Proceedings of the 3rd ACM Conference on Electronic Commerce - EC '01*, 38–47. New York, New York, USA: ACM Press. doi:10.1145/501158.501163.

Statman, Meir. 1987. "How Many Stocks Make a Diversified Portfolio?" *Journal of Financial and Quantitative Analysis* 22 (3): 353–63.

———. 2004. "The Diversification Puzzle." *Financial Analysts Journal* 60 (4): 44–53. doi:10.2469/faj.v60.n4.2636.

Surowiecki, James. 2005. *The Wisdom of Crowds*. New York: Anchor Books Random House.

Ten Hagen, S, Maarten Van Someren, and Vera Hollink. 2003. "Exploration/exploitation in Adaptive Recommender Systems." *Proceedings of Eunite 2003*, no. 634: 10–12.

The Cloud Service Measurement Initiative Consortium (CSMIC). 2013. "Service Measurement Index (SMI)." *Http://www.cloudcommons.com/about-Smi; Accessed 2013-04-12*. http://www.cloudcommons.com/about-smi.

Trapletti, Adrian, and Kurt Hornik. 2013. "Tseries: Time Series Analysis and Computational Finance."

Tsaih, Ray, Yenshan Hsu, and Charles C. Lai. 1998. "Forecasting S&P 500 Stock Index Futures with a Hybrid AI System." *Decision Support Systems* 23 (2): 161–74. doi:10.1016/S0167-9236(98)00028-1.

Tu, Jun, and Guofu Zhou. 2011. "Markowitz Meets Talmud: A Combination of Sophisticated and Naive Diversification Strategies." *Journal of Financial Economics* 99 (1): 204–15. doi:10.1016/j.jfineco.2010.08.013.

Weiser, Marc. 1991. "The Computer of the 21st Century." *Scientific American* 265 (3): 94–104.

Weng, Li-Tung, Yue Xu, Yuefeng Li, and Richi Nayak. 2008. "Exploiting Item Taxonomy for Solving Cold-Start Problem in Recommendation Making." In *Tools with Artificial Intelligence, 2008. ICTAI'08. 20th IEEE International Conference on*, 2:113–20. IEEE.

Wilcoxon, Frank. 1945. "Individual Comparisons by Ranking Methods." *Biometrics Bulletin* 1 (6): 80. doi:10.2307/3001968.

Woodside-Oriakhi, M., C. Lucas, and J.E. Beasley. 2013. "Portfolio Rebalancing with an Investment Horizon and Transaction Costs." *Omega* 41 (2): 406–20. doi:10.1016/j.omega.2012.03.003.

Xiao, Bo, and Izak Benbasat. 2007. “E-Commerce Product Recommendation Agents: Use, Characteristics, and Impact.” *Mis Quarterly* 31 (1): 137–209.

Yangui, Sami, Iain James Marshall, Jean Pierre Laisne, and Samir Tata. 2014. “CompatibleOne: The Open Source Cloud Broker.” *Journal of Grid Computing* 12 (1): 93–109. doi:10.1007/s10723-013-9285-0.

Zeithaml, Valarie A. 1985. “The New Demographics and Market Fragmentation.” *Journal of Marketing* 49 (3): 64–75. doi:10.2307/1251616.

Zhang, Dongsong, and Wei Thoo Yue. 2014. “Social Media Use in Decision Making.” *Decision Support Systems* 63. Elsevier B.V.: 65–66. doi:10.1016/j.dss.2013.08.007.

Curriculum Vitae

Jörg Gottschlich
Diplom-Wirtschaftsinformatiker

Information Systems III
Electronic Markets – Prof. Dr. Oliver Hinz
Department of Law and Economics

Hochschulstr. 1 – 64289 Darmstadt
Phone +49 (6151) 16-24445
gottschlich@emarkets.tu-darmstadt.de
http://www.emarkets.tu-darmstadt.de/team/joerg-gottschlich

Education

August 2011 – June 2016	Research Assistant and Doctoral studies Information Systems | Electronic Markets Department of Law and Economics, TU Darmstadt
October 2005 – February 2008	Business Informatics (Wirtschaftsinformatik) Main studies, Diploma Majors: System Design & Database Applications, Information Systems in Service Industries Bamberg University
September 2004 – Juni 2005	International Management (Erasmus exchange programme) University of Southern Denmark, Odense, Denmark
October 2002 – July 2004	Business Informatics (Wirtschaftsinformatik) Basic studies, Intermediate examination (Vordiplom) Bamberg University

Employment

Since August 2011	Chair of Information Systems | Electronic Markets, TU Darmstadt Research Assistant
June 2008 – Juni 2011	A.T. Kearney GmbH (Consultancy), Düsseldorf Analytics Associate
October 2005 – February 2008	addendus GmbH & Co. KG (Agency for Interim Mangement), Bamberg Student Employee for acquisition and organization Development of a CV database and scoring system
November 2003 – July 2004	Centre for commercial information systems (Cebis), Bamberg Student Assistant, IT

Awards

2015	Special Price at TU Darmstadt Innovation Competition for project “Cloudscale – Broker for Cloud Infrastructure Services”
June 2013	Claudio Ciborra Award for most innovative paper at ECIS conference, Utrecht
2008	SAS Business Intelligence Performer Award Excellent Research Paper Audience Award for Diploma Thesis

Extracurricular Activites

Music	Organ, Choir, Theater
Sports	Karate, Rowing, Running, Sailing

Frankfurt, 17.06.2016