A General Framework for Time-Aware Decision Support Systems

V. Milea[a], F. Frasincar[a], U. Kaymak[b]

[a] *Econometric Institute, Erasmus School of Economics, Erasmus University Rotterdam, P.O. Box 1738, 3000 DR Rotterdam, the Netherlands*
[b] *Information Systems Department, School of Industrial Engineering, Eindhoven University of Technology, the Netherlands*

Abstract

In this paper we present a general framework for time-aware decision support systems. The framework uses the state-of-the-art tOWL language for the representation of temporal knowledge and enables temporal reasoning over the information that is represented in a knowledge base. Our approach uses state-of-the-art Semantic Web technology for handling temporal data. Through such an approach, the designer of a system can focus on the application intelligence rather than enforcing/checking data related restrictions manually. Also, there is an increased support for reuse of temporal reasoning tools across applications. We illustrate the applicability of our framework by building a market recommendations aggregation system. This system automatically collects market recommendations from online sources and, based on the past performance of the analysts that issued a recommendation, generates an aggregated recommendation in the form of a buy, hold, or sell advice. We illustrate the flexibility of our proposed system by implementing multiple methods for the aggregation of market recommendations.

Keywords: automated trading, market recommendations, tOWL, decision support system, temporal knowledge, temporal reasoning

1. Introduction

Decision systems often rely on historical information for the formulation of a best course of action. Storing, retrieving and checking the large volumes of data and information for consistency represents one of the main challenges in building decision support systems that use historical data. Although flat representations of data used in, for example, business intelligence, provide intelligent storage and retrieval of data [5], automated inference, as needed in consistency checks, is limited in approaches based on such formalisms due to the rather inexpressive semantics of the underlying structures. Modern knowledge representation approaches provide for more finely grained semantics and additional expressiveness from a semantic perspective. From these approaches, Semantic Web [15, 31] languages such as Resource Description Framework (RDF) and RDF Schema (RDFS) [8, 13], and the Web Ontology Language (OWL) [23, 25] provide the most expressive choices when the problem of automated inference is considered. When historical data is used, some time-enabled formalism is also required. In this context, an approach such as the tOWL language [19, 21], a temporal web ontology language based on OWL, provides an appropriate formalism for the representation of time-varying knowledge.

These properties of the Semantic Web languages make them suitable for use in decision support systems, where some of the intelligence of the system is already incorporated in the representation language and does not need to be explicitly accounted for in the main system. For example, in a system where future product prices are predicted based on past prices, the restriction that a product may only have one price at any point in time can be enforced generically (through a temporal cardinality restriction), at the level of the whole knowledge base, and outside the main application. This can be achieved by using a temporal reasoner that is able to check the knowledge base for consistency, thus eliminating the need to incorporate such checks in the main system. In this way, the designer of the system can focus on the application intelligence rather than enforcing/checking data related restrictions manually. Also, there is an increased support for reuse of temporal reasoning tools across applications.

In this paper we propose a framework for designing semantic, time-aware decision support systems, based on the state-of-the-art tOWL language. The systems that we propose provide the means to efficiently store and retrieve data, and allow (temporal) inference on the represented data. Restrictions on this data can be represented in a generic way at the level of the temporal language. Different (temporal) properties of the entities in the knowledge base can also be represented, both at abstract as well as at concrete level. In this way, data-related operations are separated from the main intelligence component of the decision support system.

The framework that we introduce is deployed in a practical context. We illustrate how such semantic, time-enabled decision support systems can be used by means of an example from the financial domain. The finance area has received attention in the development of expert systems, both from a theoretical perspective, as for example in [7, 16], as well as in more practical contexts, such as [10, 24, 27]. We choose the aggregation of market recommendations as a proof-of-concept. Market recommendations are advices, in the form of indicated courses of action, issued by financial analysts, regarding the stock of a certain company. These recommendations most often materialize in *buy*, *hold*, or *sell* advices, and are issued at different times. Since multiple analysts can issue such recommendations, more often than not, at a specific point in time, a company may have recommendations issued by different analysts. When these recommendations diverge, in the sense that there is no consensus within the analyst group whether an asset should be bought, held, or sold, choosing the appropriate course of action might not be obvious. The system that we present investigates which aggregation method for market recommendations gives the best results, given the evidence from the past.

In investigating which approach provides the best results for the aggregation of market recommendations, we consider two different alternatives: a *majority voting* approach, in which we choose the recommendation to which most analysts concur, and an approach that takes into account the *analysts' past performance* when deciding the course of action with the highest expected performance. For measuring the analysts' past performance, we rely on the Sharpe ratio [32].

The example that we present contains several features that make it interesting to consider. First, the recommendations issued by analysts either have a limited validity in time, or hold until a new recommendation is issued. By relying on the tOWL language for the representation of recommendations, we can define default durations for advices, and also set as ending valid time for a recommendations the time when a new recommendation is issued by the same analyst, regarding the same company. Determining which recommendations hold at any point in time can also be achieved generically due to the timeslices representation used by the tOWL language. This allows us to determine, at any point in time, which recommendations have been issued by an analyst in the past, which in turn allows for determining the past performance of the analysts.

The outline of the paper is as follows. In Section 2 we provide an overview of the research related to the subject of this paper. The tOWL language, the basis of the framework presented in the paper, is presented in Section 3. The semantic, time-aware framework that we propose is presented in Section 4. The application, as well as the methodology we use for aggregating market recommendations is presented in Section 5. Our results and a discussion of the results are presented in Section 6. Finally, we conclude in Section 7.

2. The Temporal Dimension in Decision Support Systems

A thorough approach towards the design and evaluation of temporal expert systems is presented in [9]. The approach starts by evaluating specific characterics of expert systems and temporal applications separately, and then formulating a framework that brings both of them together. The application area that is considered for this framework consists of business problems. One of the temporal requirements that the authors formulate relates to support for a time-line view of events as well as the ability to maintain a historical repository of events, requirements that are both supported by the tOWL language. Other requirements formulated in [9] relate to being able to define and use time in different knowledge base constructs and the ability to represent temporal relationships. By relying on the tOWL language for the representation of temporal knowledge, these requirements are fulfilled due to the ability to represent temporal intervals and Allen's interval relationships [1] in the language, as well as timeslices and fluents for representing what is changing.

The importance of a temporal dimension in knowledge bases is also identified in [12] for context-dependent temporal diagnosis. The authors present an integration of Model-Based Reasoning and ontologies. Based on this framework, the authors succesfully develop a medical diagnosis system, where the domain knowledge is described in a medical ontology.

The temporal dimension in medical expert systems is also discussed in [14]. This work relies on the Temporal Utility Package (TUP), which provides, to a limited extent, some of the temporal abilities of modern, temporal Semantic Web approaches. The proof-of-concept application consists of an expert system illustrating temporal reasoning in different phases of the medical diagnostic process.

Temporal reasoning in expert systems in a more general sense is discussed in [26]. The authors develop an architecure for a temporal expert system where attribute values can be associated with time tags. The proof-of-concept consists of an expert system used for diagnosing a specific set of problems of the Hubble space telescope.

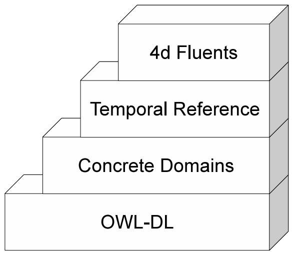

Figure 1: tOWL layer cake.

Similarly, the temporal dimension is also considered for recommender systems in [2]. The model presented is an extension of a standard Bayesian network. This extension introduces temporal nodes for the representation of states or events, and arcs for the representation of causaltemporal relationships between nodes.

Although the temporal dimension in decision support systems has been investigated to a certain extent in the literature, approaches relying on Semantic Web technologies have not yet been considered. Semantic Web languages, such as the state-of-the-art tOWL language, fulfil the domain-independent requirements formulated until now in different studies. Hence, the Semantic Web approach that we propose provides a generic mechanism for dealing with domain-independent aspects of temporal knowledge representation and reasoning in time-aware decision support systems.

3. The tOWL Language

The tOWL language [11, 18–21] is a temporal web ontology language based on the $\mathcal{SHIN}(\mathcal{D})$ description logic, which is an expressive subfragment of OWL-DL [17]. tOWL is built on top of OWL-DL, which is the current state-of-the-art ontology language and W3C standard. An overview of the different layers introduced by the tOWL language on top of OWL-DL is provided in Figure 1. The language enables the representation of time and time-related aspects, such as change. For the representation of time, the tOWL language relies on concrete domains, and enables both instant-based and interval-based representations, as well as the relations that may exist between instants and intervals (such as Allen's 13 interval relations [1] in the case of intervals). For the representation of more complex aspects, such as change, the tOWL language is designed around a 4-dimensional view of the world. In this view, the so-called timeslices are used to represent, otherwise static, OWL individuals across temporal intervals, and fluents are used to indicate what is changing. This design enables the representation of, for example, processes, and the associated state transition axioms. An example of how a complex process, e.g., a leveraged buyout process, can be represented in the tOWL language is given in [11]. The focus of the language is solely on valid time, i.e., the time when an axiom is true in the real world.

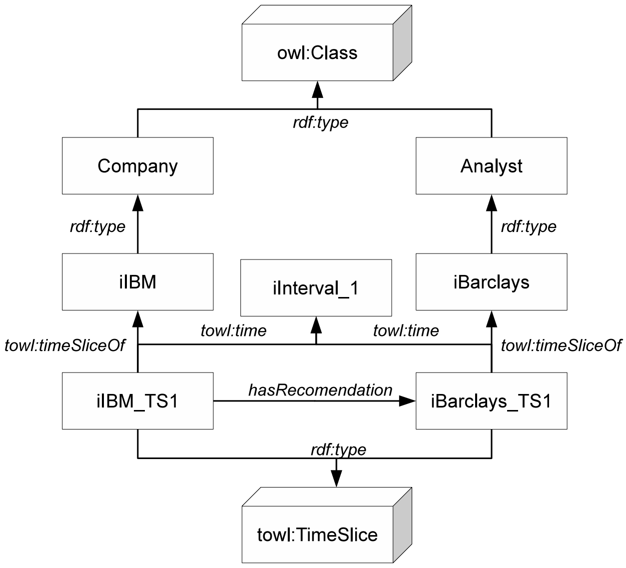

Figure 2: Representing change in tOWL.

The timeslice-based representation has the ability to determine, at any point in time, what holds true. In order to employ this representation, one has to create timeslices for the static individuals that are involved in a relation that is ephemeral in nature. For example, if one wants to represent market recommendations issued for a company, and the ontology contains static individuals that represent both the analyst issuing the recommendation, as well as the company, then timeslices have to be instantiated for both of these static concepts. Upon having done this, the two timeslices can be connected by a fluent, such as the *hasRecommendation* fluent, to indicate that, over the time interval associated with the timeslices, the two timeslices are in the *hasRecommendation* relationship. This example is illustrated in Figure 2.

In the example presented in Figure 2, two OWL classes have been defined, namely *Company* and *Analyst*. For each of these classes, one individual is instantiated, namely *iIBM*, representing the company IBM, an instance of the *Company* class, and *iBarclays*, representing the Barclays bank, an instance of the *Analyst* class. For each of these individuals, a timeslice is instantiated, namely *iIBM_TS1* and *iBarclays_TS1*, respectively. These timeslices both hold over the same interval, *iInterval1*, a consequence of the design of the tOWL language (fluents can only connect timeslices that hold over the same interval), and thus represent the static individuals with which they are associated over that interval. To denote that Barclays has a recommendation for IBM over the period denoted by *iInterval1*, we connect the two timeslices by the *hasRecommendation* fluent.

In this paper, we use the tOWL language both for the representation of static knowledge, such as company names and ticker symbols, as well as temporal knowledge, such as the interval for which an advice holds true. The tOWL knowledge is thus employed for representing all information that is considered

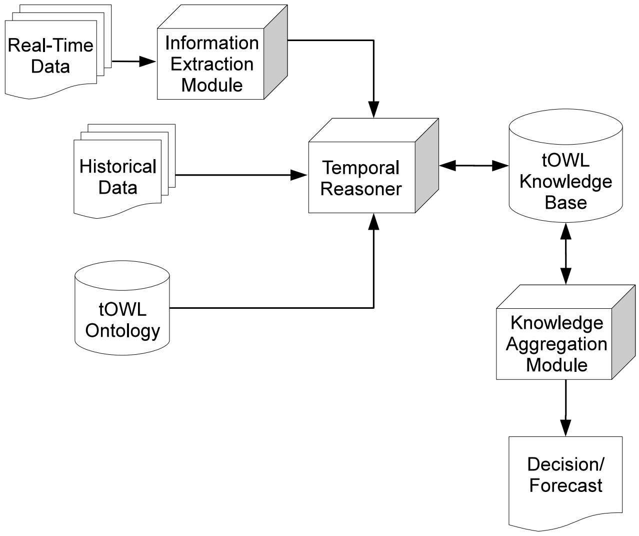

Figure 3: General architecture of the time-aware decision support system.

relevant for the system, and forms the backbone of all experiments that we perform. The tOWL formalism supports the representation of market recommendations and the time they are valid through intervals and Allen's temporal relationships between intervals. Additionally, the timeslice-based representation supports determining which advices hold true at any point in time.

4. Framework for Time-Aware Decision Support Systems

In this section we describe the general architecture of the framework for time-aware decision support systems that we propose. The different subsections discuss the design considerations and functionality of each component. Additionally, we describe the design of each component for the case study used in this paper, i.e., the aggregation of market recommendations. The general architecture of the system is presented in Figure 3.

The input of the system we propose consists of three main sources: *real-time data*, *historical data*, and a *temporal domain ontology* (designed in the tOWL language) describing the input data at an abstract level. Assuming that the real-time data is extracted from a raw source, i.e., the data is not annotated, the *information extraction module* is used to extract the relevant information from the source. All input information is fed to the *temporal reasoner* for consistency checks and the update of the *tOWL knowledge base*. The *knowledge aggregation module* uses data from the tOWL knowledge base for the generation of a forecast or recommended decision - the output of the system. Below, we discuss each of the main components in more detail.

4.1. Information Extraction Module

The purpose of the information extraction module is the retrieval of information from sources that provide data that is not annotated. The motivation behind such a component is that most information is provided in raw format, e.g., information presented as pure, not annotated text, and quick processing of this data is crucial for timely forecasts and, generally speaking, decision systems. This module can consist of various components aimed at the processing of raw data. When raw text data is considered, components such as a part-of-speech tagger can be used, as well as components that rely on patterns for the extraction of knowledge. When the textual data is only analysed at a superficial level, different content analysis components can be implemented in the information extraction module.

4.2. Temporal Reasoner

In our framework, the temporal reasoner represents the interface for populating the tOWL knowledge base. The creation of the required instances is based on a set of input sources, namely: the real-time data extracted through the Information Extraction Module, historical data, and the tOWL domain ontology that describes, at an abstract level, the domain for which the real-time data is extracted. The reasoner also ensures consistency of the knowledge base, both at a static and temporal level. An additional input to the temporal reasoner consists of the tOWL knowledge base itself, that is used for checking the consistency of the real-time data obtained through the Information Extraction Module with the current version of the knowledge base.

4.3. The Temporal Ontology and Knowledge Base

The domain for which the temporal decision support system is designed is described in at least one temporal ontology. This ontlogy provides an abstract description of the entities, the relationships that may exist between these entities and properties of these relationships. The instantiation of the temporal ontology, the temporal knowledge base, contains a description of (a part of) the studied domain at a concrete level. Additionally, through the expresiveness of the tOWL language, the knowledge base can represent how concrete entities change over time or, in case processes are represented on the knowledge base, how these processes have transtioned through different phases.

4.4. Aggregation Module

The aggregation module is aimed at using existing information for the generation of a forecast/optimal decision. The term aggregation is used here in the broad sense of a model taking various inputs and generating an output that can be used in decision-making. The inputs are stored in the tOWL knowledge base(s).

4.5. System Output

The general architecture that we propose outlines a time-aware decision support system. Thus, the system we propose is aimed at decision support, where a decision is regarded as a selection between different available courses of action.

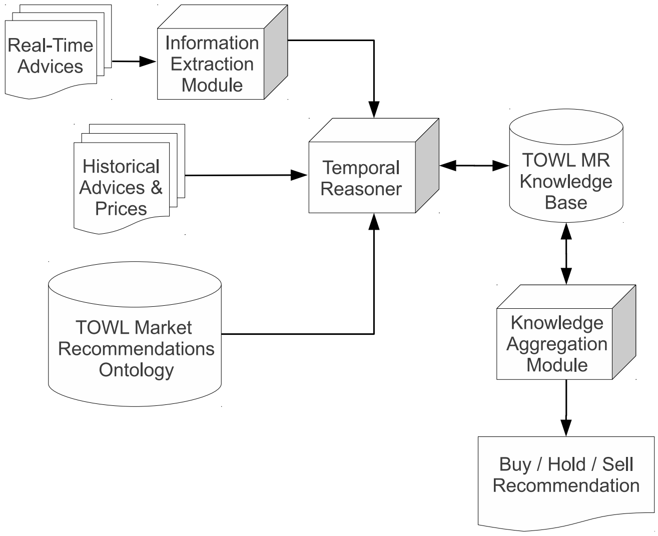

Figure 4: The market recommendations aggregation system.

5. Instantiation of the Framework

Based on the general framework presented in this paper, we can formulate an instantiation of this framework. This instance of our framework consists of a market recommendations aggregation system. For this system, the different components can be instantiated as presented in Figure 4. Here, the real-time data consists of market recommendations, i.e., advices of three different types: buy, hold, and sell. Historical data is data related to asset prices, as well as past recommendations. The tOWL ontology describes properties of recommendations as well as firms issuing the recommendations. Based on these sources, the temporal reasoner is used to update the tOWL knowledge base with new recommendations and performance data on firms. From the tOWL knowledge base, the advice aggregation module gathers and aggregates relevant data for the generation of an aggregated advice, which is also the output of the system.

5.1. Instance of the Information Extraction Module

For the application studied in this paper, we rely on Analist.nl, a Dutch language source for international market recommendations. Extracting the information contained in the advices consists of two main parts, namely: Part-of-Speech (POS) tagging and pattern-based extraction. The POS-tagger annotates every word in an advice with the part-of-speech it represents from the following eight categories: verbs, nouns, pronouns, adjectives, adverbs, prepositions, conjunctions and interjections [3]. There are different implementations available of POS-Taggers, such as Microsoft's HMM Tagger [3] and the Stanford POS-Tagger [33, 34].

Due to the fact the we use a non-English source for the advices, a POS-tagger that is able to deal with the Dutch language is used, namely TreeTagger [29], which implements the probabilistic tagging method as explained in [30].

An example of a non-tagged advice in XML as extracted from Analist.nl is provided in Figure 5, while the tagged advice is displayed in Table 1.


```
<item>
<title>RBC Capital Markets: APPLE INC. kopen.</title>
<link>http://rss.feedsportal.com/c/637/f/8254/s/1047096/story01.htm</link>
<description>(Analist.nl) Toronto - Op 6-5-2008 herhalen de analisten van RBC Capital Markets hun koopadvies voor APPLE INC. (ISIN: US0378331005 / TICKER: AAPL).Het 12-maands koersdoel voor APPLE INC. wordt opwaarts bijgesteld van 200.00 USD naar 220.00 USD. In 2006 bedroeg.. (lees verder op: www.analist.nl)</description>
<guid isPermaLink="false">http://www.analist.nl/rss.php?id=51910</guid>
</item>
```


Figure 5: RSS feed of advice *51910*.

Using the example presented in Table 1, we proceed to extract the information as follows. The first cardinal numeric value is the issuing date. The verb following the cardinal number indicates whether the advice is upgraded, downgraded, or held constant. Next, we search for the advice type, which represents words like “kopen” (buy), “houden” (hold), or “verkopen” (sell), or any of their synonyms. In case of an upgrade/downgrade, search for a second occurrence of these words “kopen” (buy), “houden” (hold) or “verkopen” (sell), or any of their synonyms. The second occurrence of the advice type indicates the new recommendation in the case of an upgrade or downgrade. The price target (the price that the asset will reach according to the analyst’s prediction) is found by searching for the first cardinal number after the word “koersdoel” (price target). If the price target is followed by the word “niet” (not) then there is no price target. In case of upgrading or downgrading the price target, the final price target will be preceded with the word “naar”. The ISIN identifier can be found by searching for the word ISIN and extracting the cardinal number following this word. The ISIN number is used for uniquely identifying a company. Finally, we extract the issuer of the advice. This is done by searching for the first occurrence of the word “van”. Since the length of the name of the analyst is unknown, all the words between the first occurrence of the word “van” after the word “analist” and the next occurrence of a verb, common noun, adjective, or adverb is chosen as the name of the broker. Extensive tests of this method provided no errors in extracting the name of the issuer.

5.2. Instance of the Temporal Reasoner

For the application used to illustrate the functionality of the proposed framework, i.e., the aggregation of market recommendations, the temporal reasoner is mostly used for determining the temporal validity of recommendations. The recommendations are assigned a default validity duration of six months, or until a new recommendation is issued. In the future, this can be made a parameter and be optimized for its best value. Thus, with the addition of new recommendations to the tOWL knowledge base, the relevant instances are checked for determining the new interval, if applicable, for which they are valid.

Table 1: Advice 51910 POS-Tagged by TreeTagger.

Word	**POS**	**Lemma**
Op	nounsg	<unknown>
6-5-2008	num__card	@card@
herhalen	verbprespl	herhalen
de	det__art	de
analisten	nounpl	analist—analiste
van	prep	van
RBC	adj	<unknown>
Capital	nounsg	<unknown>
Markets	nounpl	<unknown>
hun	det__poss	hun
koopadvies	nounpl	<unknown>
voor	prep	voor
APPLE	adj	<unknown>
INC	nounpl	<unknown>
ISIN	nounsg	<unknown>
US0378331005	nounsg	<unknown>
AAPL	nounsg	<unknown>
wordt	verbpressg	worden
opwaarts	adj	opwaarts
bijgesteld	verbpapa	bijstellen
van	prep	van
200.00	num__card	@card@
USD	adj	<unknown>
naar	prep	naar
220.00	num__card	@card@
USD	adj	<unknown>

5.3. Instance of the Temporal Ontology and Knowledge Base

For the framework proposed in this paper, we rely on the state-of-the-art tOWL language for the representation of the ontology. In the market recommendations aggregation application, we use two ontologies, namely a Financial Domain Ontology (FDO), as well as a Market Recommendations Ontology (MRO). The FDO describes financial entities and the relationships that may exist between these entities at an abstract level. The MRO is focussed on the representation of market recommendations and describes entities and the relationships amongst them in order to enable the concrete representation of market recommendations. The MRO uses the FDO for a significant part of the representation.

The FDO describes companies and is mostly focussed around their properties, such as the name, ticker, the stock exchange where the company is listed, the industry sector in which the company is active, etc. Additionally, this ontology describes analysts in terms of their unique code, name, and affiliation (where available). The recommendations in the MRO are matched to companies from the FDO. In the MRO we describe the basic properties of market recommendations, such as the analyst that issued the recommendation, the company for which the recommendation is issued, and the default duration of recommendations. Here, we assume a default validity of a recommendation to be six months, unless a new recommendation is issued by the same analyst, for the same company, within this time interval. This is relevant for the aggregation of recommendations since we need to be able to determine, at any point in time, the recommendations that hold for a company in order to generate the aggregated recommendation.

The knowledge bases for FDO and MRO contain the concrete knowledge that is available about companies, analysts, and already issued recommendations. Especially in the case of the MRO knowledge base, the temporal dimension of the tOWL language provides added value in storing and reasoning with temporal knowledge. This is mostly due to the temporal nature of recommendations, i.e., they only hold for a limited, predefined period of time (6 months), or until a new recommendation is issued. The temporal reasoner is able in such cases to adjust the ending time of a recommendation based on information that becomes available.

5.4. Instance of the Aggregation Module

For the market recommendations aggregation system, the aggregation module processes the recommendations that hold true at any point in time, and, based on the past performance of the analysts that have issued those recommendations, generates an aggregated advice in the form of buy, hold, or sell, for the asset that is being considered. The profitability of market recommendations has already been explored in finance literature. The estimation of abnormal returns by using everyday portfolio balancing based on the consensus of market recommendations is investigated in [4]. Profit can be generated by buying the most recommended stocks and selling the less favoured ones. In [6] the authors conclude that following the advices given by the broker provides a better result than following the TSE 300 or the S&P 500. In [6], [22] and [4], it is stated that investors can yield abnormal returns by following recommendations, although in [4] investors yield these high returns only when following recommendations with high consensus among the analysts. We note that the authors of [28] have done a large literature study of 250 papers. The authors suggest that no individual broker has enough information to always give correct advices. Traders will aggregate advices given by brokers and other information about a certain company to make a decision to buy, hold or sell a stock.

Previous studies have thus shown that market recommendations do have an impact on developments regarding stock prices, i.e., abnormal returns can be yielded by following stock advices. Additionally, consensus plays an important role in the lucrativeness of these advices, and a meaningful way of aggregating the individual advices might lead to improved results in terms of abnormal returns. Finally, taking into account the possible 'subjectivity' of brokers can further help improve the performance of an investment strategy based on market consensus.

In this section we outline our proposal for aggregating individual recommendations issued by analysts into a single recommendation. This aggregated recommendation takes into account the past performance of the analysts being considered. In computing this performance, we correct the achieved average return of analysts by the standard deviation of those returns. The resulting

$$\begin{array}{lcl} A_1 & : & \texttt{Buy} \\ A_2 & : & \texttt{Buy} \\ A_3 & : & \texttt{Buy} \\ A_4 & : & \texttt{Hold} \\ A_5 & : & \texttt{Hold} \\ A_6 & : & \texttt{Sell} \end{array}$$

Figure 6: Advice distribution – example

$$\begin{array}{lcl} A_1 & : & a_1, a_2 \\ A_2 & : & a_3 \\ A_3 & : & a_4 \\ A_4 & : & a_5, a_6 \\ A_5 & : & a_7 \\ A_6 & : & a_8 \end{array}$$

Figure 7: Historical recommendations – example

measure, known as the Sharpe ratio [32], gives a quantification of the performance in terms of achieved return corrected for the risk taken. The Sharpe ratio is computed for each advice type, i.e., buy, hold, and sell.

We illustrate the computation of the aggregated recommendation by means of an example. At time t, the distribution of advices presented in Figure 6 is known for company C, given the analysts denoted as A_n. Given this distribution, the goal is to compute the aggregated recommendation by taking into account the past performance of the analysts. In the past, each of the analysts has issued the advices presented in Figure 7, denoted as a_n.

In order to determine the expected performance of each advice type, the most obvious choice would be to look at the past returns of the analysts and aggregate these for each advice type. This expected performance for an advice type, say buy, can be obtained as follows in our example. Here, r_{a_n} represents the return generated by the advice a_n one day after the recommendation was issued.

$$E(R_B) = \frac{(r_{a_1} + r_{a_2} + r_{a_3} + r_{a_4})}{4} \tag{1}$$

In similar manner we obtain the expected returns $E(R_H)$ and $E(R_S)$ for the other two advice types. However, judging the performance of an advice solely in terms of generated returns paints an incomplete image, since the risk taken for obtaining these returns is not considered. We choose to measure the risk in terms of the standard deviation of the returns as follows. In the following example, we compute the risk for buy recommendations for our imaginary stock:

$$
\begin{aligned}
\sigma(R_B) \quad = \quad & \frac{1}{4}((r_{a_1} - E(R_B))^2 + (r_{a_2} - E(R_B))^2) + \\
& +(r_{a_3} - E(R_B))^2 + (r_{a_4} - E(R_B))^2)^{1/2}
\end{aligned} \tag{2}
$$

The measure of expected performance that takes into account risk, as given by the Sharpe ratio, is calculated as follows for the buy advices in our example:

$$
S_B = \frac{E(R_B)}{\sigma(R_B)}. \tag{3}
$$

In similar fashion we obtain the Sharpe ratios for the other two advice types, which we denote as S_H and S_S.

Considering only the maximum value of the Sharpe ratio in determining the aggregated advice would not take into account the number of analysts that issued recommendations for each advice type. Therefore, we use a weighted measure of performance that accounts for the number of analysts who issued an advice type, as well as the Sharpe ratio. We obtain this by multiplying each Sharpe ratio, for each advice type, with the number of analysts that issued advices of that advice type. This is computed as follows for the buy advices in our example, where n_B^t denotes the number of recommendations that hold at time t for the buy advice type:

$$
P_B = n_B^t S_B. \tag{4}
$$

Having computed this performance measure for the other two advice types, denoted as P_H and P_S, respectively, the aggregated recommendation is computed as the maximum of the three individual P values.

Generalizing this approach, we begin by computing, for each advice type, the expected return $E(R_x)$ as follows:

$$
E(R_x) = \frac{\sum_{i=1}^{m} r_{a_i}^h}{m}. \tag{5}
$$

Here x can be either buy, hold, or sell, m denotes the total number of advices issued in the past by the analysts who issued recommendation x, h denotes the time horizon being considered for the computation of the returns of the individual advices, and a_i are the individual advices being considered.

The standard deviation of the returns per advice type, $\sigma(R_x)$, is computed as:

$$
\sigma(R_x) = \left(\frac{1}{m} \sum_{i=1}^{m} (r_{a_i}^h - E(R_x))^2 \right)^{1/2}. \tag{6}
$$

The Sharpe ratio is then computed for each advice type x:

$$
S_x = \frac{E(R_x)}{\sigma(R_x)}. \tag{7}
$$

The expected performance, P_x, of each advice type as a function of its associated Sharpe ratio, S_x, and the number of advices holding at time t for advice type x, denoted as n_x^t, becomes:

$$P_x = n_x^t S_x. \quad (8)$$

Finally, the aggregated recommendation is determined by choosing the advice type with the maximum expected performance, P_x.

5.5. Application Output

For the market recommendations aggregation system, the output consists of an aggregated recommendation, in the form of either a buy, hold, or sell recommendation. This aggregated recommendation is given at a certain point in time, and holds until the system generates an aggregated recommendation that is different from the current recommendation.

6. Performance Evaluation

In this section we present the results obtained from performance analysis of different market recommendations aggregation methods by using the time-enabled decision support system described in this paper. In Section 6.1 we describe the experimental setup that stands at the basis of the experiments that we perform. The results we obtain are presented in Section 6.2. We conclude with a discussion of the results in Section 6.3.

6.1. Experimental Setup

For the experiments, we use data collected for the period January 1st, 2000 to December 31st, 2010. The data consists of all the recommendations issued for a company listed in the Dow Jones Industrial Average (DJIA) index, a set consisting of 30 companies, such as American Express, Boeing, and JPMorgan Chase.

We compute the performance of analysts that have issued a recommendation based on their past performance in the three years prior to the recommendation being considered. The performance is thus computed over a moving window of three years previous to the point in time when an aggregated recommendation is being computed. This computation takes into account all recommendations issued by the analyst for any company in any of the US markets, thus not being restricted to past performance in relation to the DJIA companies.

The aggregated recommendations are computed for every day in the dataset, but we only measure the performance of recommendations that, at time t, are different from the recommendation issued at $t-1$. In other words, although our system computes daily aggregated recommendations, we consider an advice to be issued by the system only if that advice is different from the advice issued on the previous occasion. It is only for these recommendations that we compute the performance in terms of returns for different time horizons.

The performance of recommendations is computed both in terms of returns as well as Sharpe ratios, for different time horizons. The horizons that we consider are: one day, one week, one month, half-year, and one year, respectively. The strategies that we consider are *Index*, which consists of investing in the DJ30 index with no additional strategies, *Majority voting*, where we consider the advices issued by the highest number of analysts as the aggregated recommendation, *Analyst performance*, where the aggregated advice is computed by taking into account the past performance of the analysts (measured by the

	1-day	1-week	1-month	Half-year	1-year
Index	0	-0.0002	0.0012	0.0054	0.0143
Majority voting	-0.0004	-0.0008	-0.0002	0.0141	0.0247
Analyst perf.	-0.0002	-0.0003	0.0018	0.0092	0.0171
Equally-weighted index	0.0001	0.0004	0.0014	0.0045	0.0098

Table 2: Mean returns of the different strategies

	1-day	1-week	1-month	Half-year	1-year
Index	0.0128	0.0267	0.0520	0.1151	0.1796
Majority voting	0.0270	0.0533	0.0967	0.1815	0.2644
Analyst perf.	0.0269	0.0529	0.0979	0.2201	0.3107
Equally-weighted index	0.0224	0.0490	0.0935	0.1953	0.2701

Table 3: Standard deviation of returns

Sharpe ratio), and *Equally-weighted index*, which is a variant of the DJ30 index where all companies are given equal weights. We note that the design of the system easily enables different aggregation methods to be tested. However, our focus is on illustrating the time-related capabilities of the system rather than providing the optimal aggregation method for analyst recommendations.

6.2. Performance analysis

In this section we provide an overview of the performance of the aggregated recommendations. Table 2 presents the mean returns for the four investment strategies considered, for each of the five time horizons that we use. When the two aggregation methods are compared, the majority voting method is outperformed by the aggregation method that takes into consideration the past performance of analysts, for short time horizons (1-day, 1-week, and 1-month). However, this relationship reverses for longer time horizons (half-year and 1-year). When compared with the performance of the index, the latter outperforms both methods for very short time horizons (1-day, 1-week), but is outperformed by either one or the other aggregation method for the remaining time horizons, with one exception: for 1-month majority voting performs worse than the index. Additionally, for very short time horizons the equally weighted index outperforms all aggregation methods.

In Table 3 we present the standard deviations of the returns computed for the four different investment strategies, at different time horizons. We note that, when comparing the two aggregation methods, the standard deviations for short time horizons (1-day, 1-week) are highly similar, with slighter lower values for the aggregation method based on the Sharpe ratio. At longer time horizons, the majority voting method attains lower standard deviations. However, both methods incur more risk when compared to the index, the latter displaying lower standard deviations for all time horizons.

Last, we present the performance of the different investment strategies in terms of the Sharpe ratio in Table 4. As in the case of returns, when comparing the two aggregation methods, the majority voting method is outperformed on the shorter term (1-day, 1-week, 1-month), but superior for longer time horizons (half-year and 1-year). For short time horizons, the index outperforms both aggregation methods in terms of the Sharpe ratio, which is also the case for the

	1-day	1-week	1-month	Half-year	1-year
Index	0.0062	0.0112	0.0230	0.0473	0.0793
Majority voting	-0.0157	-0.0165	-0.0024	0.0778	0.0667
Analyst perf.	-0.0082	-0.0054	0.0188	0.0420	0.0551
Equally-weighted index	0.0071	0.0095	0.0149	0.0231	0.0362

Table 4: Sharpe ratios

time horizon of one year. For the half-year interval, the majority voting method is able to outperform both the index, as well as the equally-weighted index.

6.3. Discussion

The previous section outlines the results obtained for the two aggregation methods that we consider, as well as the performance of the DJ30 index and a fictive, equally-weighted index. In terms of raw performance, measured as mean returns, the two aggregation methods are dependent in their performance on the time horizon being considered. An aggregation method that takes into account the past performance of analysts when performing the aggregation, delivers superior results for relatively short time horizons. However, for longer time horizons, a majority voting approach is superior. Despite this result, both aggregation methods are outperformed by the DJ30 index for very short time horizons (one day and one week), as well as by the equally-weighted index. However, both aggregation methods outperform the DJ30 index as well as the equally-weighted index for time horizons of half-year and one year, leading to the conclusion that the information available to analysts when issuing their recommendations has an abnormal effect only in the long term.

In terms of risk, measured as standard deviation, the index is the least risky investment strategy considered, followed by the equally-weighted index, a predictable result. Despite this, we note that the majority voting aggregation method outperforms the equally-weighted index, in terms of incurred risk, for longer time horizons (half-year and one year). The majority voting aggregation method outperforms both the index, as well as the equally weighted index, in terms of Sharpe ratio for the half-year time horizon. For shorter time horizons, the previous results based on the standard deviation where the index was less risky than the aggregation methods are also supported by the Sharpe ratio computations.

7. Conclusions

This paper describes a framework for time-aware decision support systems. The framework we propose relies on the state-of-the-art tOWL language for the representation of temporal information. We illustrate how temporal information can be used in decision support systems in a systematic, consistent way. Additionally, by relying on such an approach, the process of data storage and retrieval, as well as consistency checks on the data are separated from the main application. The added value of the paper consists of a framework for temporal decision support systems that relies on state-of-the-art Semantic Web technology for handling temporal knowledge.

We illustrate the applicability of our system in a practical context, by extracting, storing, and aggregating information related to market recommendations. By implementing different aggregation methods we demonstrate the flexibility of the proposed system. Additionally, we show how the systematic storing of knowledge by relying on the tOWL language enables different models to be deployed within the same application, models that (may) use different pieces of data from the information available within the knowledge base(s).

Although the system presented in this paper is used for an application in the financial domain, the applications of the system are not restricted to this domain. For making use of the proposed framework, one needs to define a tOWL domain ontology, the input information, the historical data, and an application engine able to process all this temporal information into a valuable output. We envision applications in the automated processing of news by news agencies, accompanied by tagging and classification of these news items based on subject, geographic area, entities involved, i.e., persons and locations.

As future work we plan to extend the proposed framework to exploit tOWL expressions directly in the information extraction phase, based on the tOWL ontology and knowledge base(s) used in the application. This can be achieved in different ways, one of which is the definition of patterns based on the the knowledge base that can directly be used by the information extraction module.

References

[1] Allen, J., 1983. Maintaining knowledge about temporal intervals. Communications of the ACM 26 (11), 832–843.

[2] Arroyo-Figueroa, G., Sucar, L., Villavicencio, A., 1998. Probabilistic temporal reasoning and its application to fossil power plant operation. Expert Systems with Applications 15 (3), 317–324.

[3] Banko, M., Moore, R. C., 2004. Part of speech tagging in context. In: The 20th International Conference on Computational Linguistics (COLING 2004). ACM, pp. 556–561.

[4] Barber, B., Lehavy, R., McNichols, M., Trueman, B., 2001. Can investors profit from the prophets? Security analyst recommendations and stock returns. The Journal of Finance 56 (2), 531–563.

[5] Batini, C., Ceri, S., Navathe, S., 1991. Conceptual database design: an entity-relationship approach. Addison Wesley.

[6] Bjerring, J., Lakonishok, J., Vermaelen, T., 1983. Stock prices and financial analysts' recommendations. The Journal of Finance 38 (1), 187–204.

[7] Blue, J., Andoh-Baidoo, F. K., 2010. Directive decision devices: Extending the reach of automation into the finance domain. Expert Systems with Applications 37 (1), 45 – 54.

[8] Brickley, D., Guha, R., 2004. RDF vocabulary description language 1.0: RDF schema. W3C Recommendation.

[9] Chinn, S., Madey, G., 1997. A framework for developing and evaluating expert systems for temporal business applications. Expert Systems with Applications 12 (3), 393–404.

[10] Falavigna, G., 2012. Financial ratings with scarce information: A neural network approach. Expert Systems with Applications 39 (2), 1784 – 1792.

[11] Frasincar, F., Milea, V., Kaymak, U., 2010. tOWL: Integrating time in OWL. In: Virgilio, R. D., Giunchiglia, F., Tanca, L. (Eds.), Semantic Web Information Management: A Model-Based Perspective. Springer, Ch. 11, pp. 225–246.

[12] Juarez, J., Campos, M., Palma, J., Marin, R., 2008. Computing context-dependent temporal diagnosis in complex domains. Expert Systems with Applications 35 (3), 991–1010.

[13] Klyne, G., Carroll, J., 2004. Resource description framework (RDF): Concepts and abstract syntax. W3C Recommendation.

[14] Kohane, I., 1987. Temporal reasoning in medical expert systems. Tech. rep.

[15] Lee, T., Hendler, J., Lassila, O., et al., 2001. The semantic web. Scientific American 284 (5), 34–43.

[16] Matsatsinis, N., Doumpos, M., Zopounidis, C., 1997. Knowledge acquisition and representation for expert systems in the field of financial analysis. Expert Systems with Applications 12 (2), 247 – 262.

[17] McGuinness, D., Van Harmelen, F., 2004. OWL web ontology language overview. W3C recommendation.

[18] Milea, V., Frasincar, F., Kaymak, U., 2008. Knowledge engineering in a temporal semantic web context. In: The Eighth International Conference on Web Engineering (ICWE 2008). IEEE Computer Society Press, pp. 65–74.

[19] Milea, V., Frasincar, F., Kaymak, U., 2012. tOWL: A temporal web ontology language. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics. 42 (1), 268–281.

[20] Milea, V., Frasincar, F., Kaymak, U., di Noia, T., 2007. An OWL-based approach towards representing time in web information systems. In: The 4th International Workshop of Web Information Systems Modeling Workshop (WISM 2007). Tapir Academic Press, pp. 791–802.

[21] Milea, V., Frasincar, F., Kaymak, U., Houben, G., 2012. Temporal optimisations and temporal cardinality in the towl language. International Journal of Web Engineering and Technology 7 (1), 45–64.

[22] Morgan, J., Stocken, P., 2003. An analysis of stock recommendations. RAND Journal of Economics 34 (1), 183–203.

[23] Motik, B., Patel-Schneider, P., Parsia, B., Bock, C., Fokoue, A., Haase, P., Hoekstra, R., Horrocks, I., Ruttenberg, A., Sattler, U., et al., 2009. OWL 2 web ontology language: Structural specification and functional-style syntax. W3C Recommendation.

[24] Nedovic, L., Devedzic, V., 2002. Expert systems in finance: a cross-section of the field. Expert Systems with Applications 23 (1), 49 – 66.

[25] Patel-Schneider, P., Hayes, P., Horrocks, I., 2004. Web ontology language (OWL) abstract syntax and semantics. W3C Recommendation.

[26] Perkins, W., Austin, A., 1990. Adding temporal reasoning to expert-system-building environments. IEEE Expert 5 (1), 23–30.

[27] Rada, R., 2008. Expert systems and evolutionary computing for financial investing: A review. Expert Systems with Applications 34 (4), 2232 – 2240.

[28] Ramnath, S., Rock, S., Shane, P., 2006. A review of research related to financial analysts forecasts and stock recommendations. Unpublished working paper, Georgetown University.

[29] Schmid, H., 1995. Treetagger – a language independent part-of-speech tagger. Tech. rep.

[30] Schmid, H., 1997. Probabilistic part-of-speech tagging using decision trees. In: Jones, D., Somers, H. (Eds.), New Methods in Language Processing. Studies in Computational Linguistics. UCL Press, London, GB, pp. 154–164.

[31] Shadbolt, N., Hall, W., Berners-Lee, T., 2006. The semantic web revisited. Intelligent Systems, IEEE 21 (3), 96–101.

[32] Sharpe, W., 1966. Mutual fund performance. The Journal of Business 39 (1), 119–138.

[33] Toutanova, K., Klein, D., Manning, C., Singer, Y., 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In: The 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology (HLT-NAACL 2003). Association for Computational Linguistics, pp. 173–180.

[34] Toutanova, K., Manning, C., 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In: The 2000 Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora (EMNLP/VLC-2000). Association for Computational Linguistics, pp. 63–70.