Index Terms— **business indicator, BInd use case, data warehouse functionality, actor aggregation, estimated from**

I. Introduction

In recent years, data warehouse requirements engineering has emphasized the determination of the information contents of the warehouse To Be. Indeed, all work in data warehouse requirements engineering known to the authors considers this aspect only. Consider the three life cycles described in [19]. The main task in the data base driven [9] and ER driven [10] life cycles is to restructure data bases and ER diagrams respectively to determine the required facts and dimensions. Goal oriented approaches [2, 3, 9, 18, 19] explore system/organizational goals and determine star schemas. The recent proposal [17] for building a data warehouse for formulation of rules for policy enforcement also focuses on the information aspect. [20] has introduced the notion of a target. Targets participate in two hierarchies, the relevance and fulfillment hierarchies. It has been shown that these hierarchies lead to determination of the information to be kept in the data warehouse to be. In [17, 20] the process of arriving at star schemas has been split into two parts (i) an 'early information' part where the information relevant to decision making is discovered and (ii) a 'late' part where the discovered information is structured as facts and dimensions.

Now, in concentrating on information discovery and structuring, data warehouse requirements engineering **de-emphasizes** an investigation into its functional aspect: what functions should be built for different stakeholders. We propose that one source of data warehouse functionality is business indicators. Computation of indicators in the absence of well-defined mathematical functions was considered in [1] but no attempt was made to elicit the set of functions needed in a data warehouse. Even though performance indicator systems like Performance Pyramid [15] and Balanced Scorecard [11] for developing performance indicators exist, defining the right indicators remains a major issue [5, 7, 22].

Work on requirements engineering for data warehouse functionality starts once indicators are finalized by business people. The first step in the requirements engineering task is to elicit these. However, business indicators are dispersed [22] in an organization and involve many people. Thus, a method for discovering stakeholders and their indicators is crucial for elicitation.

In this paper, we propose an elicitation mechanism for business indicators and then consider the needed functionality. For this purpose, we use the notion of a target hierarchy defined in [20]. A target is an association of a set of **indicators** with a **work aspect**. A work aspect is a work unit or a work area.

Our requirements engineering process is in two steps:

a) Identifying stakeholders and eliciting business indicators from them,

b) Determining indicator- sub-indicator hierarchy, thereby identifying functional requirements for computing indicators.

In the first step of requirements engineering process, a set of <S, BInd> pairs is identified where S is a stakeholder and BInd is the business indicator (computed by S) together with its arguments. In the second step, we construct use case diagrams so as to visualize the needed functionality for obtaining the business indicator. Our use case diagrams include two features in addition to those found in UML use case diagrams. These are (a) actor aggregation and (b) the *estimated from* relationship between use cases.

In the next section we present our Indicator model and use it for eliciting indicators. Section 3 deals with representation of BInd use case diagrams. Section 4 contains an example to

2 Eliciting business indicators

We define a business indicator as a function with arguments $x_1, x_2, \ldots x_n$, where x_i may be any of the following:

i) business indicator BInd, for example, BInd_College_Success Index = f(Engg. Success index, Science success index) – this consists of two arguments which are business indicators themselves.

ii) non- functional argument or information, other than business indicator. For example, Top5=f(marks_all_students) – this consists of a list of marks obtained by all students.

iii) function that is not a business indicator itself. For example, DeptSuccess_Index = f(count (passed_students), count (total_students)) – this consists of count function applied on a list of passed students and a list of total students respectively.

Based on this, we describe our Indicator Exchange Model and use this model as the basis for eliciting indicators and the indicator hierarchy.

2.1 The Indicator Exchange Model

Our indicator exchange model is based on the Organization Structure Model, OSM. According to OSM [16], an organization consists of organization units that may be internal or external. The OSM guidelines for defining an organizational unit are that it must be persistent and be a formal association of persons. Thus, departments, divisions, committees, can all be units. A position is a role in an organization unit and there may be more than one person holding it. Assignments link people to positions. A person may have multiple assignments, linking them to multiple positions. Business Functions are OSM meta concepts for defining what an organizational does.

Fig. 1 shows our Indicator exchange model. This model

A non-empty set of *work* is associated to a *work aspect*. This *charged with* association is 1: n as shown. There are two kinds of work aspect, work unit and work area. The *work unit* of Fig. 1 is a recognized association of *positions*, reflecting the organizational structure. Thus *work unit* ISA *work aspect*. It can be seen that *work unit* corresponds to the OSM notion of organization unit. *Work area* is a recognized association of *positions* charged with a certain organization task. The Accounts Department of an organization is an example of a **work unit**: it is a set of positions and defines a structural organization unit. An example of **work area** is Tax Reform: it is a set of positions defined to reform tax computation and deduction.

We define two ternary relationships, *informs* and *reports to*, both of which are between a pair of *positions* and the *work* these positions carry out. The relationship *reports to* says that *position* P_i reports to *position* P_j, the *indicator* for a *work* responsibility and activity. Likewise, the relationship *informs* says that *position* P_i informs *position* P_j, the *indicator* for a *work* responsibility and activity.

As shown in Fig. 1, *business indicator* is an association class with attribute *indicator*. Since, more than one indicator may be reported/informed, it is a multi-valued attribute.

The difference between *reports to* and *informs* is that whereas the former captures indicator exchange between formal positions, the latter captures indicator exchange between positions that are not bound by the reporting structure of the organization chart.

As we will see, these two relationships

- Provide us a good starting point for indicator exchange elicitation. Specifically, through these relationships we can focus on *positions* as sources of indicators.
- Form a basis for capturing the dispersed business indicators of an organization referred to in the Introduction.

```
                BIndE_i is an argument of
                BIndS_i
            End if
        End for
      End if
    End for
    Elicit any other argument of BIndS_i
  End for
End
```

The above algorithm is applied to each element of IPS also.

In our example we start with A from RPS. We do not find any such element in SentBInd, which means that A does not send business indicators to other positions. We move to the next element, B. B sends $\{BI_1, BI_2\}$ to A. We now look at ExpectsBInd for B and find that it contains $\{BI_1, BI_2\}$. Since both BI_1 and BI_2 are self generated by B, information required to compute these must be available to B itself. Further, these could only be ii) or iii) above. This information is elicited from B. Let the computation information elicited be $BI_1=f(I_{11}, I_{12})$ and $BI_2=f(I_{21})$.

Now, consider the next position D. D sends $\{BI_5, BI_6, BI_7\}$ to A and receives $\{BI_8, BI_9\}$ from E, $\{BI_{10}\}$ from F and $\{BI_{11}, BI_{12}\}$ from G. The computation information elicited is $BI_5=f(BI_8, BI_9)$; $BI_6=f(BI_{10}, I_{61}, I_{62})$ and $BI_7=f(BI_{11}, BI_{12}, I_{71})$. Now the business indicators for the remaining positions E, F and G are further examined and information is elicited. The resulting functions are $BI_8=f(I_{81}, I_{82})$; $BI_9=f(I_{91})$; $BI_{10}=f(I_{101}, I_{102})$; $BI_{11}=f(I_{111}, I_{112})$; $BI_{12}=f(I_{121})$.

The hierarchies for our business indicators are shown in Figure 3. As seen there are five hierarchies, for BI_1, BI_2, BI_5, BI_6 and BI_7 respectively.

a. **Actor aggregation**: If an indicator goes into estimating the value of another, then the stakeholder of the former is in the hierarchy of the latter. Thus, a stakeholder 'is part of' another stakeholder and we introduce the notion of aggregate actors in BInd Use Case diagrams. Notice that UML provides for actor specialization/generalization and not for actor aggregation.

b. ***Estimated from* relationship between use cases**: Since an indicator is estimated from another sub-indicator, the BInd use case diagram must contain one use case for the composite indicator and another for the sub-indicator. We now need to introduce a new relationship, *estimated from*, between use cases.

To understand the need for (b) above, consider the two UML relationships, extend and include:

a) **Extend -** specifies that a use case extends the base use case. The base use case may stand alone, but under certain conditions, its behaviour may be extended by the other.

b) **Include -** Specifies that the base use case explicitly incorporates the behaviour of another use case at a location specified by the base. The included use case never stands alone, but is only instantiated as part of some larger base that includes it.

The *estimated from* is a relationship between a base use case and an *estimated from* use case. Both use cases are capable of independent existence and have actors associated with them. The actor of the base use case interacts with the system and so does the actor of the *estimated from* use case. In this sense, the *estimated from* is different from UML's *include* relationship. It is also different from *extend* in that there is no extension of any behaviour. The indicator produced by the

online, denominados hipervídeos, especificando índices a determinadas zonas del vídeo y el acceso a recursos externos o actividades desde determinados puntos del vídeo. Algunas de estas actividades son exámenes de tipo test que se evaluarán automáticamente si es posible. Los resultados pueden utilizarse como retroalimentación del alumno o para análisis del profesor usando otros servicios externos. Finalmente, se proporciona una técnica de almacenamiento masivo de los resultados en las imágenes de un vídeo que puede ser consultado para extraer dichos datos.
En el desarrollo de este sistema se han utilizado, entre otras, técnicas de metamodelado, lenguajes de marcado XML y OpenCV.

Abstract

Online education and especially MOOC (Massive Open Online Courses), use massively videos whose design need new techniques that integrates naturally learning resources and activities of the students in the video. The VideoData system facilitates the creation of quality content for online education, called hypervideo, specifying indexes to certain areas of the video and access to external resources or activities from certain video points. Some of these activities are multiple choice exams that are automatically evaluated if it's possible. Results can be used as student feedback or analyzed using other external services. Finally, it is provided a technique for massive storage of the results in video frames, which can be consulted to extract data. In the development of this system they have been used, among others,

1. Motivación

La educación on-line [1] es un tipo de educación a distancia que hace uso de Internet y de las tecnologías de la información y la comunicación con el objetivo de facilitar el proceso de aprendizaje.

Una modalidad de educación online, que se ha extendido en los últimos años, es el uso de plataformas MOOC [2]. Estas siglas proceden del inglés Massive Open Online Course y sirven para designar cursos masivos, abiertos y online. Sin embargo, diferentes problemas están surgiendo en su desarrollo, como son el contenido de baja calidad, una alta tasa de abandono, problemas de evaluación y la gestión de datos masivos que resultan de estos cursos [14,15].

En la educación online, y especialmente en los MOOC, el uso de vídeos docentes resulta el mejor tipo de documento para llegar de forma directa al alumno. Por ello es frecuente que el diseño de un curso MOOC gire alrededor de una colección de vídeos a partir de los cuales se proponen actividades a los alumnos. Se necesitan nuevas técnicas en el diseño de vídeos docentes, que integren de forma natural los recursos de aprendizaje, las actividades y las pruebas de evaluación de los alumnos dentro del vídeo. Este nuevo tipo de recursos los denominaremos hipervídeo.

VideoData[] es un sistema integrado que permite diseñar cursos on-line basados en hipervídeos. Para ello se parte de un modelo educativo en el que destacamos las fases de creación de contenidos, evaluación y análisis y almacenamiento de resultados.

En este trabajo destacaremos dos aspectos de VideoData. El primer aspecto ha sido definir las funcionalidades que definen un hipervídeo y los parámetros

RESEARCH IN THE SCHOOLS
ix-xxxvi

Editorial: Evidence-Based Guidelines for Avoiding the Most Common APA Errors in Journal Article Submissions

Anthony J. Onwuegbuzie, Julie P. Combs, John R. Slate, and Rebecca K. Frels
Sam Houston State University

In this editorial, we provide evidence-based guidelines to help authors avoid committing APA errors. Specifically, we provide guidelines for adhering to APA style using findings from Combs, Onwuegbuzie, and Frels' (2010) mixed analysis of 110 manuscripts submitted to Research in the Schools over a 6-year period. Combs et al. identified the 60 most common APA errors grouped into 14 themes. We contend that an efficient way for authors to learn APA style is to focus initially on these common errors and error themes. Further, we contend that these errors provide useful starting points for persons who teach APA style. Finally, authors of the APA Publication Manual might use this information to determine which rules and guidelines to emphasize.

As co-editors and first-round copyeditors of *Research in the Schools* (John R. Slate and Anthony J. Onwuegbuzie), outgoing editor and associate editor of *Educational Researcher* (Anthony J. Onwuegbuzie and Julie P. Combs, respectively), recent guest editor of the *International Journal of Multiple Research Approaches* (Anthony J. Onwuegbuzie), editorial assistant/production editor of *Research in the Schools* (Rebecca K. Frels), and reviewers for and editorial board members of numerous journals, we have observed the difficulties that many authors have experienced in conforming to the guidelines specified without an adequate knowledge of the *Publication Manual* style. Unfortunately, this inadequacy likely makes the transition from doctoral student to beginning author to emergent scholar more difficult. Indeed, over the years, we have observed that some of our reviewers (i.e., editorial board members) have extremely low tolerance for APA errors. Thus, it is clear that authors who submit manuscripts to journals wherein APA style is required would benefit from becoming as familiar as possible with the *Publication Manual*.

Unfortunately, during the last 9 years, authors