2009

Requirement Analysis for Enterprise Information Systems - Developing an Ontological Meta-Model for Zackman Framework

Zhuozhi Chen
Heriot-Watt University, zc20@macs.hw.ac.uk

Rob Pooley
Heriot-Watt University, r.j.pooley@hw.ac.uk

REQUIREMENT ANALYSIS FOR ENTERPRISE INFORMATION SYSTEMS - DEVELOPING AN ONTOLOGICAL META-MODEL FOR ZACKMAN FRAMEWORK

Completed Research Paper

Zhuozhi Chen
Department of Computer Science
Heriot-Watt University
Edinburgh, Scotland
United Kingdom
zc20@macs.hw.ac.uk

Prof. Rob Pooley
Department of Computer Science
Heriot-Watt University
Edinburgh, Scotland
United Kingdom
r.j.pooley@hw.ac.uk

Abstract

An enterprise information system distinguishes itself from other types of software as it is developed to facilitate the operation of an organization hence its requirement reflects its strategies, plans, organizations, processes, marketing etc. We believe that the requirements in the form of domain knowledge acquired in the early stage of system development can be organized and modeled in an Enterprise Architecture. Zachman Framework is one of the most widely used Enterprise Architectures. However, in the original version of the Zachman Framework, there is neither a rigorous meta-model nor a well-defined sequence in which to instantiate the cells, which prevents it from being used practically during the requirement engineering phase of an enterprise information system project. To improve such a situation we develop a conceptual meta-model for the Zachman Framework by adapting and integrating the Bunge-Wand-Weber ontology and the Enterprise Ontology. Based on this meta-model, various requirement acquisition processes can be formulated by specifying a sequence to traverse the meta-model graph and instantiate its nodes and edges. In this paper we present such a process, suitable for an enterprise system development project of a particular situation.

Keywords: Enterprise Information Systems, Requirement Modeling, Enterprise Architecture, Zachman Framework, Meta-Model, BWW Ontology, Enterprise Ontology

Introduction

Enterprise information systems account for a large portion of software deployed world wide. They perform vital roles in supporting the efficient running of almost all enterprises. As a result of this they are of major importance among all different types of software, e.g. real time system, embedded software systems etc. Requirement engineering (RE) research has produced a large number of requirement analysis and modeling techniques. Many of these RE methods are model-based techniques, e.g. goal-directed requirement acquisition [28, 10, 11, 1, 2], scenario-based requirement analysis [42, 43, 44], viewpoint-oriented requirement elicitation [26, 27, 38, 39], goal-scenario coupling [34, 36, 3, 30, 9], object-oriented RE [4]. However, few of these methods are specially tailored for enterprise information systems. On the contrary, most existing requirement analysis techniques are claimed to be able to cope with any domain and used in any type of software development. We think this makes these methods difficult to adopt in real life project, as requirement analysts have to demonstrate great flexibility. They must interpret, for themselves, some concepts embedded in these methods because some of them can have very different meaning in particular domains. For example, "goals", as a crucial notion in goal-oriented RE, can have very

different interpretations in the requirement analysis of an enterprise information system and in real-time aircraft navigation software. Therefore we advocate that requirement elicitation should be more domain specific.

In this paper we focus on the enterprise information system domain and develop an RE method that is specifically tailored for it and hence more efficient than generic RE methods. The paper is structured as follows. The Zachman Framework [51, 40] is briefly reviewed in the first section. BWW Ontology [48, 49, 50] and Enterprise Ontology [15, 47], which are ontologies at different levels, are introduced in the second section. Then a conceptual meta-model for the Zachman Framework is presented; this is adapted and integrated using both the BWW Ontology and the Enterprise Ontology. Finally one specific requirement elicitation process suitable for a particular situation is described.

Enterprise Architecture and Zachman Framework

Enterprise information systems distinguish themselves from other types of software in that they are developed to facilitate the operation of an organization and, hence, reflect the knowledge of the enterprise's structure, strategies, plans, organizations, people, activities, processes, resources, business rules, external relations etc. The complete computational representation of all such information can be called an Enterprise Model or Enterprise Architecture. A significant portion of requirement engineering activity is about acquiring, eliciting and modeling such information; in other words, in Enterprise Modeling or in building an Enterprise Architecture. To develop such an Enterprise Architecture for a specific enterprise, some meta-architecture should be used to facilitate communication and provide terminology. This kind of meta-architecture is widely known as an Enterprise Architecture Framework. We believe that requirement engineering activities for enterprise information system should be organized under an Enterprise Architecture Framework. Various enterprise architecture frameworks have been proposed from both industry and academia, e.g. the Zachman Framework, ARIS, TOGAF [56], FEAF, C4ISR, DoDAF, MoDAF Framework and many more.

Among them the Zachman Framework is the earliest and most widely used. It was first proposed by J. Zachman in [51] as a 3 column/5 row matrix, derived by drawing an analogy between information systems architecture and classical architecture. The five rows represent the different perspectives of the stakeholders involved in the planning, conception, designing, building and using of the information systems of the organization. These five different perspectives each have three sets of names which can be Scope/Contextual/Planner, Business Model/Conceptual/Owner, System Model/Logical/Designer, Technology Model/Physical/Builder and Detailed Representations/Out-of-Context/User. The three resulting columns were later extended to six columns [40], representing DATA/What, FUNCTION/How, NETWORK/Where, PEOPLE/Who, TIME/When and MOTIVATION/Why respectively, to describe different aspects of the enterprise and its systems. The most recent version of the Zachman Framework is shown in Figure 1. An enlarged version can be found online at [52].

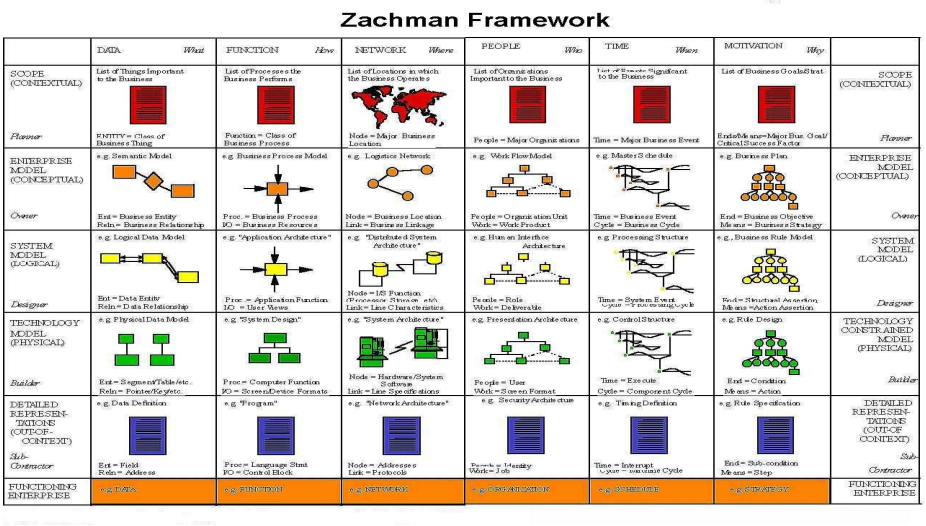

Figure 1. Zachman Framework

Zachman Framework has been widely accepted in industry. However, the original version is difficult to use as practical guidance for enterprise system development without some extension, especially during the requirement engineering phase. Reasons for this are summarized below:

- There is no rigorous meta-model for each cell.
- There is no clearly defined dependency or relationship between the cells.
- There is no recommended sequence to create models for each cell. Therefore it does not explicitly give any guidance on the requirement elicitation process.

Many commercially available Enterprise Architecture modeling tools have included a version of the Zachman Framework in their products. Such EA modeling tools as IBM Telelogic Rational System Architect [57] and Sparx Enterprise Architect [58] have adopted and developed extensive modeling notations for each cell. There are also cross-cell linkages between meta-models of different cells. However, due to its commercial nature, the underlying meta-model remains unavailable to the public.

To develop a meta-model for the Zachman Framework, we need to look to other fields for inspiration. In this paper we show how ontology, which is originally a subject in philosophy, can help us.

Ontology

Ontologies have been increasingly used in Artificial Intelligence and Information System. In Artificial Intelligence, ontologies are used as a tool for knowledge representation and natural language processing. In the information systems discipline, two different ways of applying ontologies are found. One is similar to ontologies in knowledge engineering, in that it is used as a technology to formally represent agreed domain semantics. The other approach uses ontology in its original philosophical sense, as a benchmark for evaluating the expressiveness of conceptual modeling techniques.

BWW Ontology: An Upper Level Ontology

Wand and Weber are among the first researchers to initiate the use of ontology theories in information system analysis and design. They [48, 49, 50] adapted and extended an ontology presented by Bunge [5, 6], and applied it to the conceptual modeling of information systems. The result is widely referred to as the Bunge-Wand-Weber (BWW) model. The BWW model consists of the representation model, the state-tracking model, and the good decomposition model. A representation model defines a set of constructs that are thought to be necessary and sufficient to describe the structure and behavior of the real world. A complete list and explanation of all the constructs in the BWW representation model can be found in their work [49, 50]. Here in Table 1 we only list the constructs that are essential to understand our example.

Table 1. Selected Constructs of BWW Ontology (Taken from Rosemann and Green, 2002)

Ontological Construct	Explanation
Thing	A thing is the elementary unit in the BWW ontological model. The real world is made up of things. Two or more things (composite or simple) can be associated to form a composite thing.
Property In general In particular Intrinsic Non-binding mutual Binding mutual Hereditary	Things possess properties. A property is modeled via a function that maps the thing onto some value. For example, the attribute ‘‘weight’’ represents a property that all humans possess. In this regard, weight is an attribute standing for a property in general. If we focus on the weight of a specific individual, however, we would be concerned with a property in particular. Other properties are properties of pairs or many things. Such properties are called mutual. Non-binding mutual properties are those properties shared by two or more things that do not ‘‘make a difference’’ to the things involved; for example, order relations or equivalence relations. By

Emergent Attributes	contrast, binding mutual properties are those properties shared by two or more things that do ‘‘make a difference’’ to the things involved. A property of a composite thing that belongs to a component thing is called a hereditary property. Otherwise it is called an emergent property. Some properties are inherent properties of individual things. Such properties are called intrinsic. Attributes are the names that we use to represent certain properties of things (normally abstract properties).
Class	A class is a set of things that can be defined via their possessing a characteristic property.
Kind	A kind is a set of things that can be defined only via their possessing two or more properties.
Coupling Binding Mutual Property	Two things are said to be coupled (or interact) if one thing acts on the other and vice versa. Furthermore, those two things are said to share a binding mutual property (or relation); that is, they participate in a relation that ‘‘makes a difference’’ to the things.
System	A set of things is a system if, for any bi-partitioning of the set, couplings exist among things in the two subsets.
System composition	The things in the system are its composition.
System environment	Things that are not in the system but interact with things in the system are called the environment of the system.

They also enunciated two evaluation criteria. First, a modeling grammar (set of modeling symbols and their construction rules) is deemed *ontologically complete* if it contains constructs that enable it to model any real-world phenomenon in the target domain that might be of interest to an information system. Where this is not true the resultant model is *ontologically incomplete*.

Second, a modeling grammar is deemed *ontologically clear* if each of its constructs has a one-to-one correspondence with one of the BWW ontological constructs. If not, there can be three situations:

1) One grammatical construct may map to two or more ontological constructs. This situation is called *construct overload*.

2) Two or more grammatical constructs may map to one ontological construct. This situation is called *construct redundancy*.

3) A grammatical construct may not map to any ontological construct. This situation is called *construct excess*.

BWW ontology has been widely used as the benchmark ontology for evaluating the expressiveness of various conceptual modeling languages.

Enterprise Ontology: A Lower Level Domain Ontology

Unlike an upper level ontology, very few enterprise ontologies have been developed, although there are abundant Enterprise Modeling methodologies. One of the reasons for a lack of comprehensive enterprise ontologies is that such ontologies must be able to represent various concepts of time, activity, actor, resource, organization, market, strategy etc. Many of these concepts are rather abstract which makes it difficult to derive definitions that can be linked to an upper level ontology. Furthermore, these concepts must be integrated to support reasoning.

We have found two projects worldwide that have built a complete enterprise ontology. One is the TOVE Project at the Enterprise Integration Laboratory, University of Toronto [53]. The goal of the TOVE project is to create an ontology that has the following characteristics [15]:

- provides a shared terminology for the enterprise that every application can jointly understand and use

- defines the meaning (semantics) of each term in as precise and as unambiguous as possible a manner using first-order logic
- implements the semantics in a set of PROLOG axioms that enable TOVE to automatically deduce the answer to many commonsense questions about the enterprise
- defines a symbology for depicting a term, or the concept constructed thereof, in a graphic context

Currently the TOVE project has produced ontologies of Activity [23], Resource [14], Organization [17], Product and Requirement [29], Quality [24] and Cost [45].

The other is The Enterprise Project at Artificial Intelligence Applications Institute at the University of Edinburgh [54]. The Enterprise Ontology they developed was initially informal, consisting of terms and definitions in natural language. They later converted it into a formal ontology encoded in Ontolingua [22], an online distributed collaborative ontology editing environment. The syntax and semantics of Ontolingua definitions are based on a notation and semantics for an extended version of first-order predicate calculus called Knowledge Format Interchange Format (KIF) [18]. Their experiences during the conversion were recorded in [46]. A complete list of terms is listed in Table 2. Their definitions expressed in natural language can be found in [47]. The Ontolingua version is held online in the Library of Ontologies maintained by Knowledge Systems Lab (KSL) at Stanford University [55].

We adopt the Enterprise Ontology developed by University of Edinburgh in this paper rather than the one developed by TOVE project for the following reasons:

- The former has been formally defined using an extended version of first-order logic (KIF, [18]).
- The former is integrated while the latter is fragmented.

Table 2. Terms of Enterprise Ontology

ACTIVITY	ORGANISATION	STRATEGY	MARKETING	TIME
Activity	Person	Purpose	Sale	Time Line
Activity Specification	Machine	Hold Purpose	Potential Sale	Time
Execute	Corporation	Intended Purpose	For Sale	Time Point
Executed Activity Specification	Partnership	Purpose-Holder	Sale Offer	
T-Begin	Partner	Strategic Purpose	Vendor	
T-End	Legal	Entity	Objective	Actual Customer
Pre-Condition	Organizational Unit	Vision	Potential Customer	
Effect	Manage	Mission	Customer	
Doer	Delegate	Goal	Reseller	
Sub-Activity	Management Link	Help Achieve	Product	
Authority	Legal Ownership	Strategy	Asking Price	
Activity Owner	Non-Legal Ownership	Strategic Planning	Sale Price	
Event	Ownership	Strategic Action	Market	
Plan	Owner	Decision	Segmentation Variable	
Sub-Plan	Asset	Assumption	Market Segment	
Planning	Stakeholder	Critical Assumption	Market Research	
Process Specification	Employment Contract	Non-Critical	Brand	

		Assumption		
Capability	Share	Influence Factor	Image	
Skill	Shareholder	Critical Influence Factor	Feature	
Resource		Non-Critical Influence Factor	Need	
Resource Allocation		Critical Success Factor	Market Need	
Resource Substitute		Risk	Promotion	
			Competitor	

Conceptual Meta-Model of the Zachman Framework

Our conceptual modeling approach has a similar hierarchy to the KAOS approach [28, 10, 11], which is illustrated in Figure 2.

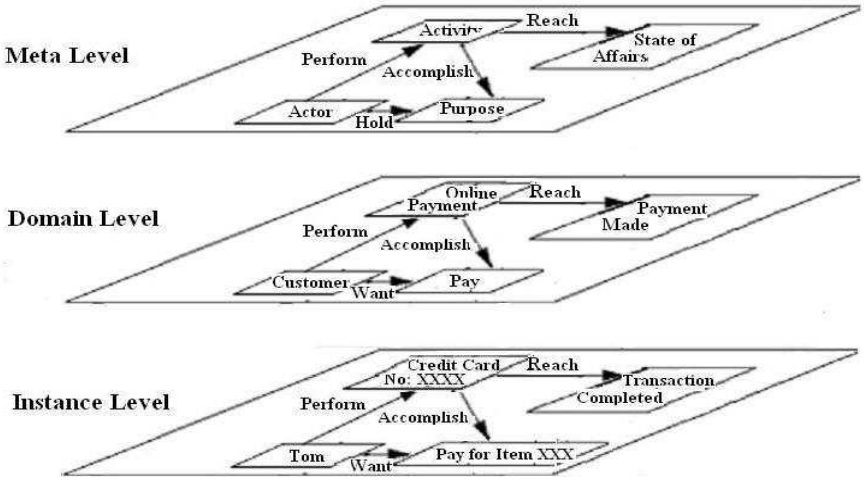

Figure 2. Conceptual Modeling Hierarchy

The meta level is composed of meta-concepts and meta-relationships, both of which are domain-independent abstractions. Meta-relationships connect meta-concepts semantically. For example, there can be several meta-relationships between meta-concept *Actor* and meta-concept *Activity*, e.g. *perform*, *capability*, *authority*, etc. Unlike the KAOS project, in our meta-model, there is no *meta-attribute* or *meta-constraint*.

The domain level is composed of domain specific instances of meta-concepts and meta-relationships. For example, in an enterprise domain, a *customer* is an instance of meta-concept *Actor*, and *make payment* is an instance of meta-concept *Activity*. There can be several relationships between them, e.g. a customer makes a payment or a customer is capable of making a payment or a customer is authorized to make a payment. These are instances of meta-relationships *perform*, *capability* and *authority* respectively.

The instance level is composed of particular instances of domain level concepts and relationships. For example, a customer named Tom makes payment for a book he purchased online using his credit card no XXXX.

To define meta-concepts and meta-relationships for our meta-model, we refer to the constructs in BWW Ontology and terms defined in Enterprise Ontology. The reason we use two ontologies is because neither of them is ontologically complete to model all six columns of the Zachman Framework. In other words, they are ontologically incomplete when used alone. We will see this clearly after we have mapped their constructs to the columns in the Zachman Framework.

Merge BWW Ontology and Enterprise Ontology

Before we can build an integrated meta-model, we need first to merge the two ontologies. Although the two ontologies are at different levels, there are synonyms in them. We merge the two ontologies by merging the synonyms as one single meta-concept. We denote such meta-concepts by adding a prefix BWW to their BWW construct name and keep their original Enterprise Ontology name. For example, in BWW the most fundamental construct is THING, while in Enterprise Ontology it is named ENTITY. We denote such a meta-concept as ENTITY (BWW THING). After merging the synonyms, we achieve a meta-model graph as illustrated in Fig. 3. To differentiate, we use red rectangles for meta-concepts derived from BWW ontology. We denote meta-relationships using rectangles in shade connected to the lines that connect meta-concepts.

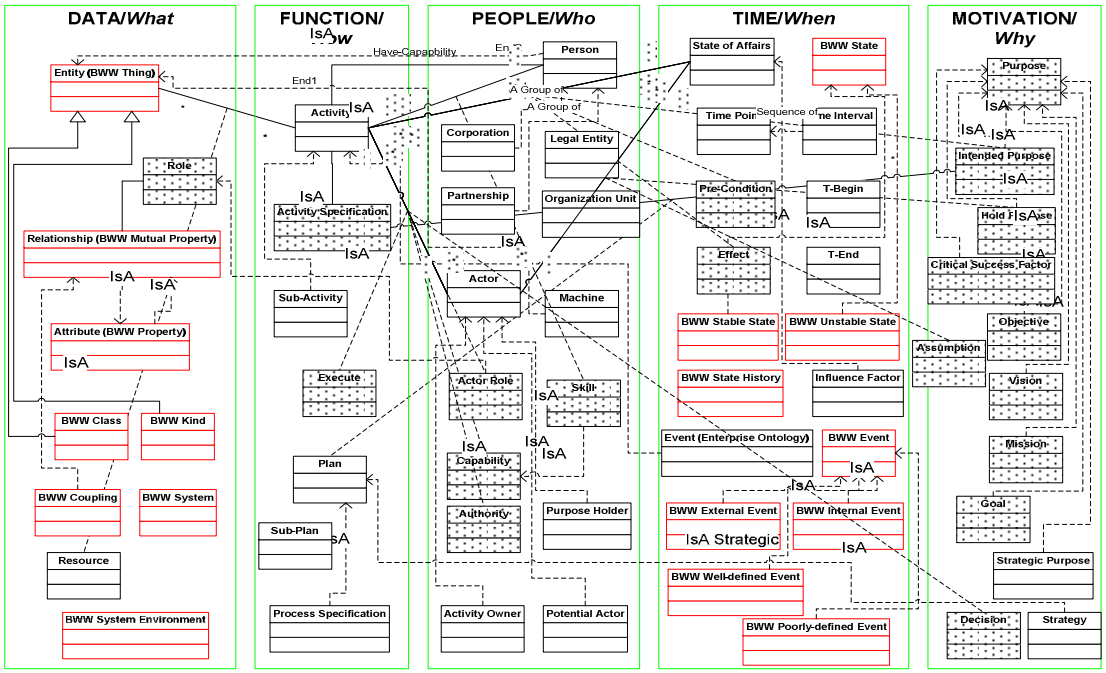

Figure 3. Conceptual Model-Model for Zachman Framework Adapted from BWW Ontology and Enterprise Ontology

Classify Concepts according to What/How/Where/Who/When/Why

The meta-concepts and meta-relationships in the meta-model graph are analyzed to see which English question word they correspond to. We observe several interesting points here:

- There are neither BWW constructs nor Enterprise Ontology terms corresponding to NETWORK/Where.
- BWW Ontology constructs mainly fall into DATA/What and TIME/When. There are no BWW constructs in FUNCTION/How, PEOPLE/Who and MOTIVATION/Why.
- Enterprise Ontology constructs spread among all five columns except NETWORK/Where.

Hypotheses can be derived for the expressiveness of BWW Ontology and Enterprise Ontology from the observations made above.

Firstly, the lack of BWW Ontology constructs or Enterprise Ontology terms that correspond to cells in the NETWORK/Where column may suggest both BWW Ontology and Enterprise Ontology are incapable of representing spatial information. Many of today's enterprises are distributed geographically with offices all over the

world. As a result their enterprise systems are distributed geographically as well. Technology advancement in distributed computing also makes today's enterprise systems more architecturally distributed. This requires spatial information to be elicited and modeled during the requirement engineering phase. To overcome the incompleteness of both ontologies, a Spatial Ontology [21, 41] should be considered and added.

Secondly, the lack of BWW Ontology constructs in FUNCTION/How, PEOPLE/Who and MOTIVATION/Why may suggest BWW Ontology is not capable of representing dynamic phenomena in the real world and differentiating "living organism" from "non-living thing", e.g. person and business objects in enterprise. Since this is not our focus in this paper, we will not discuss it further. However these observations support our initial claim that neither BWW Ontology nor Enterprise Ontology alone can be used as meta-model of Zachman Framework.

A Requirement Elicitation Process Based on Meta-Model of Zachman Framework

As mentioned in the introduction, a significant portion of the requirements of an enterprise information system acquired during the early stage of RE is the knowledge of the enterprise's structure, strategies, plans, organizations, people, activities, processes, resources, business rules, external relations etc. Such knowledge is composed of concepts and relationships, which are domain specific instances of the meta-concepts and meta-relationships. Therefore the requirement elicitation process can be seen as a defined sequence in which to traverse the meta-model graph to acquire instances of its nodes and edges. Depending on the differing situation of different system development projects, the process/sequence may be different. In this paper we present one particular process as an example which is suitable in the following situation.

An enterprise initiates a new strategy or changes an existing strategy, which in turn requires changes in its business process and information system. This can be achieved by developing a new information system or changing the existing information system. Requirements of the new information system or requirements that have to be added to the existing information system need to be analyzed and modeled. The requirement elicitation method we propose for such circumstance is described by the following seven steps:

1) Acquire High Level Goal/Strategy Tree model – MOTIVATION/Why
2) Extract STATE OF AFFAIRS from the Goal-Strategy Tree model, and build Master Schedule by adding TIME LINE/TIME POINT to STATE OF AFFAIRS – TIME/When
3) Draw Organizational Chart – PEOPLE/Who
4) Model Enterprise Data in E-R Diagram at semantic/conceptual level – DATA/What
5) Reduce High Level Goal/Strategy in Goal/Strategy Tree model until Strategy becomes operationalizable, model operationalizable Strategy using Business Process Model – FUNCTION/How
6) Analysis each PROCESS to identify concerned ACTOR ROLES and ENTITY, Model the interaction between ACTOR ROLES and ENTITY as Scenario/Use Case
7) Use scenario-oriented requirement analysis techniques to continue the requirement acquisition process

If we look at these in greater detail, we can note:

Step 1: Acquire High Level Goal/Strategy Tree model – MOTIVATION/Why

Because a new strategy is initiated or an existing strategy has changed, our RE process has to start from the top right corner of the Zachman Framework, the highest level of the MOTIVATION/Why column. The constructs in the meta-model that correspond to the MOTIVATION/Why column are STRATEGY, OBJECTIVE and GOAL, etc. First, write down a list of Business Goals/Strategies of enterprise/organization in natural language, according to the definition of GOAL and STRATEGY in Enterprise Ontology. Then transform the list into a Goal-Strategy Tree model as Zachman Framework recommends for the Enterprise Model of the MOTIVATION/Why column of, as illustrated in Figure 4.

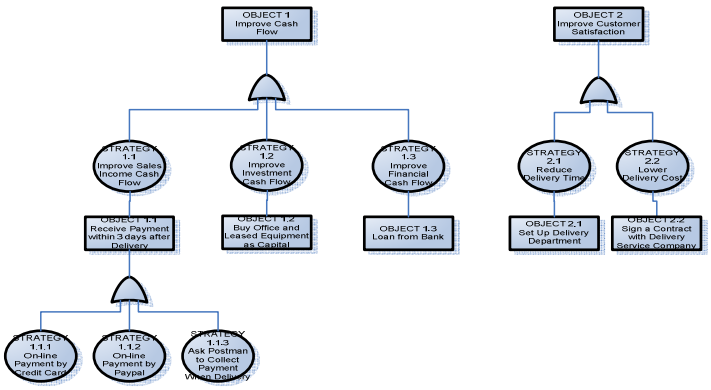

Figure 4. Sample model for Business Plan (Enterprise Model of MOTIVATION/Why column of Zachman Framework)

Step 2: Extract STATE OF AFFAIRS from the Goal-Strategy Tree model, and build Master Schedule by adding TIME LINE/TIME POINT to STATE OF AFFAIRS – TIME/When

In Enterprise Ontology, we have the definitions of terms given below:

- STRATEGY is defined as a PLAN to Achieve a STRATEGIC PURPOSE.
- PLAN is defined as an ACTIVITY SPECIFICATION with an INTENDED PURPOSE.
- INTENDED PURPOSE is a Relationship between an ACTIVITY SPECIFICATION and a STATE OF AFFAIRS where EXECUTION of the ACTIVITY SPECIFICATION will result in fully or partially ACHIEVING the STATE OF AFFAIRS.
- STRATEGIC PURPOSE is a PURPOSE of "strategic" importance.
- T-BEGIN and T-END are the two TIME POINTS that define the TIME INTERVAL over which an ACTIVITY is done.
- PRE-CONDITION is a STATE OF AFFAIRS required to be true in order for the ACTIVITY to be performed. The requirement may be specified to hold immediately before T-BEGIN, immediately before T-END, or throughout the whole TIME-INTERVAL.
- EFFECT is a STATE OF AFFAIRS that is brought about (i.e. made true) by the ACTIVITY. The EFFECT may be specified to hold immediately after T-BEGIN, immediately after T-END, or throughout the whole TIME INTERVAL.

By connecting above definitions, we can derive that:

- A STRATEGY is an ACTIVITY SPECIFICATION to fully or partially ACHIEVE a STATE OF AFFAIRS that is at STRATEGIC level. We use S, A and T to denote STRATEGY, ACTIVITY SPECIFICATION and STATE OF AFFAIRS. Then we have S = {A, T}.
- A STRATEGIC PURPOSE is a STATE OF AFFAIRS that the corresponding STRATEGY tries to achieve. In other words, a STRATEGIC PURPOSE is the EFFECT of the corresponding STRATEGY.

We compare these definitions with the Goal/Strategy Tree Model. We can see that the Objective nodes (square) are PURPOSES (of STRATEGIC level or lower level) which equals to a STATE OF AFFAIRS. The Strategy nodes (circle) are ACTIVITY SPECIFICATIONs at different levels.

Now that we have instantiated the first two cells of MOTIVATION/Why column, we want to move onto the TIME/When column. The most important constructs in the meta-model that correspond to cells in the TIME/When column are STATE OF AFFAIRS, STATE and EVENT. At this initial stage only STATE OF AFFAIRS is meaningful. We extract any STATEs OF AFFAIRS from the Goal/Strategy model and insert them into the Master Schedule by assigning them a T-BEGIN and a T-END, as shown in Figure 5.

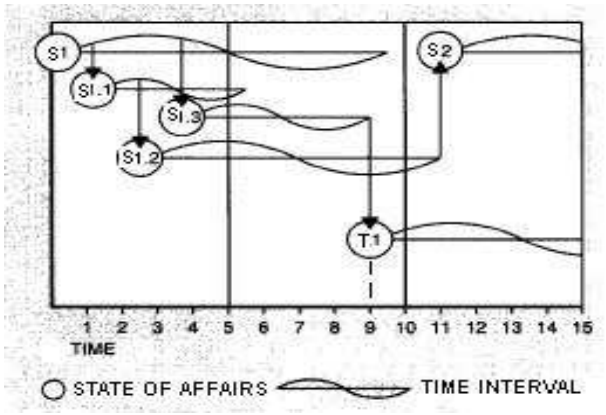

Figure 5. Master Schedule (Enterprise Model of TIME/When column of Zachman Framework)

Step 3: Draw Organizational Chart – PEOPLE/Who

The most important constructs that correspond to the PEOPLE/Who column are ORGANIZATION UNIT and ACTOR ROLES. To model such information, an organization chart can be used. An organization chart is a diagram that shows the structure of an organization and the relationships and relative ranks of its parts and positions/jobs. This kind of diagram usually already exists in some business documents. An example is given in Figure. 6. Our approach requires that at the leaf level the node should be ACTOR ROLES as defined in Enterprise Ontology, e.g. Customer, Sales Clerk and Accountant, etc.

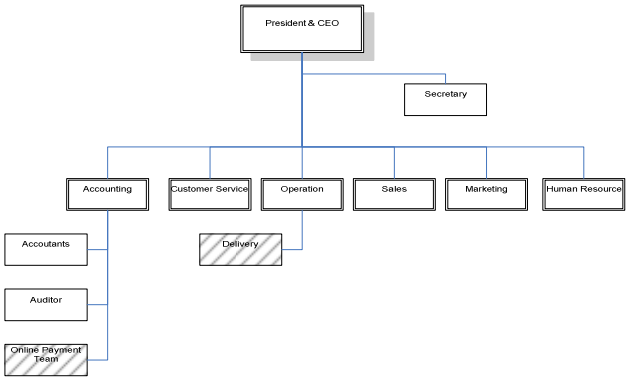

Figure 6. Organizational Chart (Enterprise Model of PEOPLE/Who column of Zachman Framework)

Step 4: Model Enterprise Data in E-R Diagram at semantic/conceptual level – DATA/What

There is abundant literature in both academic and industry on this topic [7, 8, 37]. Therefore we do not repeat it here.

Step 5: Reduce High Level Goal/Strategy in Goal/Strategy Tree model until Strategy become operationalizable, model operationalizable Strategy using Business Process Model – FUNCTION/How

Now we have elicited and modeled requirements of four columns, MOTIVATION/Why, TIME/When, PEOPLE/Who and DATA/What. We want to move on to the FUNCTION/How column. The most important construct that corresponds to the TIME/When column is ACTIVITY SPECIFICATION. We propose a method that elicits ACTIVITY SPECIFICATION by reducing STRATEGY.

There are two ways to reduce a STRATEGY. The first is to decompose the corresponding STATE OF AFFAIRS into a set of STATE OF AFFAIRS with AND relationships. For example, cut down 10% cost is a STATE OF AFFAIRS, which can be further decomposed to two separate STATE OF AFFAIRS: cut down 20% administrative cost and 5% production cost, denoted as T = T1 ^ T2.

Sometimes a STATE OF AFFAIRS cannot or is difficult to further decompose. The other way to decompose a STRATEGY is to find an alternative ACTIVITY SPECIFICATION and STATE OF AFFAIRS which in effect achieve the same result as the STRATEGY being decomposed. For example, a STRATEGIC GOAL to reach 80% of market share can be achieved by developing more new products and cutting down costs by 10%.

We want to know when the reduction process can finish, so we need to define the concept of a Strategy being Operationalizable. Before we do this, two concepts need to be differentiated: *State of Affairs* and *State* of *aThing*. The definition for them comes from our two different ontologies, Enterprise Ontology and BWW Ontology, because each of them has only defined one of the concepts.

- STATE OF AFFAIRS in the Enterprise Ontology is a situation that consists a set of RELATIONSHIPS between particular ENTITIES and it can be said to hold, or be true (and conversely to not hold or to be false). In first-order logic, any STATE OF AFFAIRS can be formally represented by a syntactically valid sentence, or formula (i.e. S1^S2^S3). Strictly speaking, to formally represent a STATE OF AFFAIRS, is to formally specify the syntax of a first-order logic sentence. Fortunately, this is already formalized in KIF, so there was no need to re-define this from scratch.
- STATE of a THING in the BWW Ontology is the vector of values for all attribute functions of the thing.

When a Strategy is Operationalizable, the STATE OF AFFAIRS which it is going to achieve is actually a STATE of one single ENTITY instead of a STATE OF AFFAIRS of several ENTITIES. The STRATEGY now is an ACTIVITY SPECIFICATION at such a low level that it is a business process. We name an Operationalizable Activity Specification a PROCESS.

Step 6: Analyse each PROCESS to identify concerned ACTOR ROLES and ENTITY, Model the interaction between ACTOR ROLES and ENTITY as Scenario/Use Case

First we analyze which ACTOR ROLES identified from Step 3 are involved in a PROCESS identified from Step 5. Similarly we find out which ENTITIES from Step 4 are involved in the processes. Then ENTITIES are analyzed to see if they are within the boundary of the system.

- If ENTITIES are outside the boundary of the system, then the PROCESS is a material process, but not an information process. A material processes relates human tasks that are rooted in the physical world. Such tasks include, moving, storing, transforming, measuring, and assembling physical objects, e.g. moving cargo into a warehouse. An information process usually is needed to reflect any such changes that need to be recorded in our information system, e.g. update Inventory Item in either a paper based inventory catalogue or a warehouse information system. If such an information process has not been identified yet, it needs to be added to the list of PROCESS.
- If ENTITIES are within the boundary of the system, then the PROCESS is an interaction taking place between ACTOR ROLES and the system; a use case and scenario should be developed in the next step.

Step 7: Use scenario-oriented requirement analysis techniques to continue the requirement elicitation process

From this step we can use various existing scenario-oriented RE techniques [42, 43, 44] to continue the requirement elicitation process.

Related Work

As mentioned in the second section, there are many Enterprise Architecture Frameworks. Many of them have methodologies that specify the sequence of developing different models, for example, TOGAF.

TOGAF (The Open Group Architecture Framework) has been developed by the Architecture Forum of the Open Group and continuously evolved since the mid-1990s. TOGAF is based on four architecture domains: Business Architecture, Application Architecture, Data Architecture and Technical Architecture. The TOGAF Architecture Development Method (ADM) provides a tested and repeatable process for developing architectures. The ADM includes establishing an architecture framework, developing architecture content, transitioning, and governing the realization of architectures. All of these activities are carried out within an iterative cycle of continuous architecture

definition and realization that allows organizations to transform their enterprises in a controlled manner in response to business goals and opportunities.

Although our method is closely related to the TOGAF ADM, the goals and scopes are completely different. TOGAF ADM is a complete methodology for developing a whole enterprise architecture, which will include the set of all applications that are deployed in the enterprise. Our work focuses on developing a requirement elicitation process for one enterprise application or a group of related applications.

A similar approach to requirement acquisition is the KAOS approach [28, 10, 11]. However their meta-model is not based on an ontology and their approach is declared to be able to be applied in any domain. This could cause inconsistency during the requirement acquisition process as the instantiation of the meta-model is difficult to have a consistent interpretation.

RML [19, 20] and TELOS [31] use knowledge representation approaches for modeling requirements of information systems. However, their approach is based on a formal language instead of a conceptual modeling method. Such languages are rarely used according to a recent survey [32]. Besides, their focus is on requirement specification instead of requirement elicitation.

Pereira and Sousa [33] identified the flexibility of the Zachman Framework as a negative issue. To deal with this they proposed a method which defines the sequence of filling up each cell with a top-down and incremental approach. They particularly defined a new concept called an "anchor cell" which is the cell in the FUNCTION/How column of each row. They claim that all the other cells in the same row have the "anchor cell" as their base. They also present a tool developed for the purpose of supporting the Zachman Framework concepts. Although they developed a sequence to instantiate cells, they did not suggest any meta-model or modeling grammar.

Kingston and Macintosh [25] suggested several modeling techniques for different perspectives of the Zachman Framework. But they neither explicitly mention relationships existing between those modeling grammars nor do they specify a sequence to acquire models using these modeling techniques.

Rosemann's initial investigation [35] has shown some issues when apply BWW ontology to enterprise system requirement. One of the issues is the gap between constructs of modeling grammars and terms found in enterprise system requirement. His claim is supported by our observation that there are no BWW constructs in FUNCTION/How, PEOPLE/Who and MOTIVATION/Why columns while Enterprise Ontology constructs spread among all five columns except NETWORK/Where.

Conclusion

In this paper we propose a conceptual meta-model for the Zachman Framework. The meta-model is made of constructs from two different ontologies, BWW Ontology and Enterprise Ontology. We firstly integrate the two ontologies by merging synonyms. Then we classify concepts according to What/How/Where/Who/When/Why. We derive some hypotheses from the results. Based on the meta-model, we further develop a specific requirement elicitation process to sequentially traverse the meta-model graph to acquire domain specific instances of the meta-concepts and meta-relationships. This process is suitable for a particular situation that a new strategy or a change to an existing strategy has caused need to develop new enterprise information system.

References

1. Anton, A. "Goal-Based Requirement Analysis", in *Proceedings of the 2nd International Conference on Requirements Engineering (ICRE '96)*, Colorado Springs, CO, USA, April 15-18, 1996, pp. 136-144.
2. Anton, A. and Potts, C. " The Use of Goals to Surface Requirements for Evolving Systems", in *Proceedings of the 20th International Conference on Software Engineering*, Kyoto, Japan, April 19-25, 1998, pp. 157-166.
3. Anton, A., Carter, R., Dangino, A., Dempster, J. and Siege, D. "Deriving Goals from a Use-Case Based Requirements Specification", *Requirements Engineering* (6:1), 2001, pp. 63-73.
4. Bailin, S. "An Object-Oriented Requirements Specifications Method", in *Communications of the ACM* (32:5), 1989, pp. 608-623.
5. Bunge, M. *Treatise on Basic Philosophy. Vol. 3, Ontology I: The Furniture of the World*, Readel, Boston, MA, 1977.
6. Bunge, M. *Treatise on Basic Philosophy. Vol. 4, Ontology II: A World of Systems*, Readel, Boston, MA, 1979.

7. Chen, P. P. S. "The Entity-Relationship Model - Toward a Unified View of Data", *ACM Transactions on Database Systems* (1:1), 1976, pp. 9-36.
8. Chen, P. P. S. (Ed.) *Proceedings of 1st International Conference on the Entity-Relationship Approach to System Analysis and Design*, Los Angeles, CA, 1979.
9. Cockburn, A. "Structuring Use Cased with Goals", *Journal of Object-Oriented Programming*, Sep/Oct, 1997, pp. 35-40, and Nov/Dec, 1997, pp. 56-62.
10. Dardenne, A., Fickas, S. and Lamsweerde, A. "Goal-directed Concept Acquisition in Requirement Elicitation", in *Proceedings of the 6th International Workshop on Software Specification and Design*, Como, Italy, October 25-26, 1991, pp. 14-21.
11. Dardenne, A., Lamsweerde, A. and Fickas, S. "Goal-directed Requirement Acquisition", *Science of Computer Programming*, 20, 1993, pp. 3-50.
12. Darimont, R. and Lamsweerde, V. "Formal Refinement Patterns for Goal-Driven Requirements Elaboration", in *Proceedings of the 4th ACM SIGSOFT Symposium on Foundations of Software Engineering*, San Francisco, CA, October 16-18, 1996, pp. 179-190.
13. Degen, W., Heller, B, Herre, H. and Smith, B. "GOL: Toward An Axiomatized Upper-Level Ontology", in *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems*, Ogunquit, ME, October 17-19, 2001, pp. 34-46.
14. Fadel, F., Fox, M., Gruninger, M. "A Generic Enterprise Resource Ontology", in *Proceedings of the 3rd Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprise*, Morgantown, WV, April, 1994, pp. 117-128.
15. Fox, M. "The TOVE Project Towards a Common-Sense Model of the Enterprise", in *Proceedings of the 5th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, 1992, pp. 25-34.
16. Fox, M. and Grüninger, M. "Enterprise Modeling", *AI Magazine*, (19:3), 1998, pp. 109-121.
17. Fox, M., Barbuceanu, M., Gruninger, M. and Lin, J. "An Organization Ontology for Enterprise Modelling", *Simulating Organizations: Computational Models of Institutions and Groups*, 1998, pp. 131-152.
18. Genesereth, R., Fikes, E. "Knowledge Interchange Format, Version 3.0 Reference Manual", *Technical Report Logic-92-1*, Computer Science Department, Stanford University, 1992.
19. Greenspan, S., Mylopoulos, J. and Borgida, A. "Capturing More World Knowledge in the Requirements Specification", in *Proceedings of the 6th International Conference on Software Engineering*, Tokyo, Japan, 1982, pp. 225-234.
20. Greenspan, S., Mylopoulos, J. and Borgida, A. "On Formal Requirements Modeling Languages: RML Revisited", in *Proceedings of the 6th International Conference on Software Engineering*, Sorrento, Italy, 1994, pp. 135-147.
21. Grenon, P. and Smith, B. "SNAP and SPAN: Towards Dynamic Spatial Ontology", *Spatial Cognition & Computation: An Interdisciplinary Journal* (4:1), 2004, pp. 69-104.
22. Gruber, T. "A Translation Approach to Portable Ontology Specifications", *Knowledge Acquisition* (5:2), 1993, pp. 199-220.
23. Gruninger, M. and Fox, M. "An Activity Ontology for Enterprise Modeling". Submitted to *AAAI-94*, 1994.
24. Kim, H. and Fox, M. "An Ontology of Quality for Enterprise Modeling", in *Proceedings of 4th Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprises*, Berkeley Springs, WV, April 20-22, 1995, pp. 105-116.
25. Kingston, J. and Macintosh, A. "Knowledge Management Through Multi-Perspective Modelling: Representing and Distributing Organizational Memory", *Knowledge-Based Systems*, 13, 2000, pp. 121-131.
26. Kotonya, G. and Sommerville, I. "Viewpoints for Requirement Definition", *Software Engineering Journal* (7:6), 1992, pp. 375-387.
27. Kotonya, G. and Sommerville, I. "Requirements Engineering with Viewpoints", *Software Engineering Journal* (11:1), 1996, pp. 5-18.
28. Lamsweerde, A., Dardenne, A., Delcourt, B. and Dubisy, F. "The KAOS Project: Knowledge Acquisition in Automated Specification of Software", in *Proceedings of AAAI Symposium Series, Track: Design of Composite Systems*, Stanford University, Palo Alto, CA, March, 1991, pp. 59-62.
29. Lin, J. Fox, M. and Bilgic, T. "A Requirement Ontology for Engineering Design", *Concurrent Engineering*, (4:3), 1996, pp. 279-291.
30. Liu, L. and Yu, Eric. "From Requirements to Architectural Design - Using Goals and Scenarios", in *Proceedings of 1st International Workshop: From Software Requirements to Architectures (STRAW 2001)*, Toronto, Canada, May 14, 2001, pp. 22-30.

31. Mylopoulos, J., Borgida, A., Jarke, M. and Koubarakis, M. "Telos: Representing Knowledge about Information Systems", ACM Transactions on Information Systems (TOIS) (8:4), October, 1990, pp. 325-362.
32. Neill, C. and Laplante, P. "Requirement Engineering The State of the Practice", *IEEE Software* (20:6), 2003, pp. 40-45.
33. Pereira, C. and Sousa, P. "A Method to Define an Enterprise Architecture using the Zachman Framework", in *Proceedings of 19th Annual ACM Symposium on Applied Computing, Session: Organizational Engineering (OE)*, Nicosia, Cyprus, March 14-17 2004, pp. 1366-1371.
34. Rolland, C., Souvey, C. and Achour, C. "Guiding Goal Modeling Using Scenarios", *IEEE Transactions on Software Engineering*, (24:12), 1998, pp. 1055-1071.
35. Rosemann, M., Vessey, I., Weber, R. and Wyssusek, B. "On the Applicability of the Bunge–Wand–Weber Ontology to Enterprise Systems Requirements", in *Proceedings of the 15th Australasian Conference on Information Systems*, Sydney, Nov 29-Dec 2, 2004.
36. Rolland, C., Grosz, G. and Kla, R. "Experience With Goal-Scenario Coupling In Requirements Engineering", in *Proceedings of the 4th IEEE International Symposium on Requirements Engineering (RE'99)*, Limerick, Ireland, June 7-11, 1999, pp. 74-81.
37. Scheer, A. and Hars, A. "Extending Data Modeling to Cover the Whole Enterprise", *Communications of the ACM* (35:9), 1992. pp. 166-172.
38. Sommerville, I. and Sawyer, P. "Viewpoints: Principles, Problems and a Practical Approach to Requirement Engineering". *Annals of Software Engineering* (3:1), 1996, pp. 101-130.
39. Sommerville, I., Sawyer, P. and Viller, S. "Viewpoints for Requirement Elicitation: A Practical Approach", in *Proceedings of the 3rd International Conference on Requirements Engineering (ICRE '98)*, Colorado Springs, CO, April 6-10, 1998, pp. 74-81.
40. Sowa, J. and Zachman, J. "Extending and Formalizing the Framework for Information Systems Architecture", *IBM Systems Journal* (31:3), 1992, pp. 590-616.
41. Spaccapietra, S., Cullot, N., Parent, C. and Vangenot, C. "On Spatial Ontologies", in *Proceedings of the VI Brazilian Symposium on GeoInformaticas (GEOINFO 2004)*, Campos do Jordao, Brazil, 2004.
42. Sutcliffe, A. "Scenario-based Requirement Analysis", *Requirements Engineering* (3:1), 1998, pp. 48-65.
43. Sutcliffe, A. and Ryan, M. "Experience with SCRAM, a Scenario Requirements Analysis Method", in *Proceedings of the 3rd International Conference on Requirements Engineering (ICRE '98)*, Colorado Springs, CO, April 6-10, 1998, pp. 164-171.
44. Sutcliffe, A., Maiden, N., Minocha, S. and Manuel, D. "Supporting Scenario-based Requirements Engineering", *IEEE Transaction on Software Engineering* (24:12), 1998, pp. 1072-1088.
45. Tham, D., Fox, M. and Gruninger, M. "A Cost Ontology for Enterprise Modeling", in *Proceedings of the 3rd Workshop on Enabling Technologies: Infrastructures for Collaborative Enterprise*, Morgantown, WV, April, 1994, pp. 197-210.
46. Uschold, M. "Converting an Informal Ontology into Ontolingua: Some Experiences", *Workshop on Ontological Engineering*, Budapest, Hungary, 1996, pp. 1-17.
47. Uschold, M., King, M., Moralee, S., and Zorgios, Y. "The Enterprise Ontology", *The Knowledge Engineering Review* (12:1), 1998, pp. 31-89.
48. Wand, Y. and Weber, R. "An Ontological Model of an Information System". *IEEE Transactions on Software Engineering* (16:11), 1990, pp. 1281-1291.
49. Wand, Y. and Weber, R. "On the Ontological Expressiveness of Information Systems Analysis and Design Grammars", *Journal of Information Systems* (3:4), 1993, pp. 217–237.
50. Wand, Y. and Weber, R. "On the Deep Structure of Information Systems", *Information Systems Journal* (5:5), 1995, pp. 203-223.
51. Zachman, J. "A Framework for Information Systems Architecture", *IBM Systems Journal* (26:3), 1987, pp. 276-295.
52. http://www.zachmaninternational.com/
53. http://www.eil.utoronto.ca/enterprise-modelling/TOVE/
54. http://www.aiai.ed.ac.uk/project/enterprise/
55. http://www.ksl.stanford.edu/software/ontolingua/
56. http://www.opengroup.org/togaf/
57. http://www.telelogic.com/products/systemarchitect/
58. http://www.sparxsystems.com/products/mdg/tech/zachman/