A Possible Metric for Semantic Similarity of Object-Oriented Analysis Models and Its Use to Estimate the Predictability of Object-Oriented Analysis Methods

Davor Svetinovic
Masdar Institute of Science and Technology
Abu Dhabi, United Arab Emirates
davor@mit.edu

Daniel M. Berry, Michael W. Godfrey, Nancy A. Day
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
{*dberry, migod, nday*}*@uwaterloo.ca*

Abstract

Ideally, the results of a mature engineering process are predictable, i.e., independent applications of the process produce results that are the same within certain controlled limits. We suggest estimating the predictability of domain models produced using an object-oriented analysis method by comparing how semantically similar and complete are the domain models produced by independent analysts applying the method to the same problem. We introduce a possible measure of semantic similarity and evaluate it by using it to compare the predictability of two object-oriented analysis methods applied to 65 independently specified domain models of a large Voice-over-IP system and its supporting information management system.

1. Introduction

As Matulevičius and Heymans [1] observe for languages for producing requirements engineering artifacts, "It is difficult to judge the quality of a language only [sic] from reading the documentation." The same is true for methods for using languages. The usual evaluation of a language or method for specific purposes consists of matching its features against what is needed of a language or method for the given purposes. However, in the last analysis, the true test of the suitability of a language or a method for specific purposes is how good for these purposes are the artifacts that the language or method is used to produce [2], [3]. This paper proposes and evaluates one possible measure of the quality of languages and methods for the modeling done in object-oriented analysis (OOA). The metric, *predictability*, is calculated on the artifacts produced during OOA.

In one previous work [4], we discussed the difficulties we observed among students learning and performing traditional OOA within a large course project, leading to the conclusion that traditional OOA is not as simple as the folklore claims it is. Here, traditional OOA is a scenario-based OOA, such as that described by Larman [5]. In response to these difficulties, we introduced in a second work, hereinafter called the UCUM paper [6], an OOA method called the Use Case Unification Method (UCUM). In UCUM, a functional model, in the form of a statechart [7], of the entire domain of a computer-based system (CBS) is built from its use cases (UCs) before decomposing the domain model into objects. UCUM is an informal variant of the formal methods presented in the works of Glinz [8]; Whittle and Schumann [9]; and Harel, Kugler, and Pnueli [10]. From a *qualitative evaluation* of UCUM's effectiveness in attacking these difficulties, we concluded that UCUM is straightforward to learn and to apply, that it provides significant help in handling the difficulties we observed in building a domain model, that it helps students produce better models than previously, but that it is not the cure for all domain-model building difficulties [6]. After observing these qualitative differences, we searched for a quantitative way to observe the same differences.

The current paper explores a different, quantitative measure of the quality of the traditional OOA and UCUM methods, namely one based on the repeatability of a method, i.e, on the predictability of the results of the method. Each of science and engineering depends on repeatable and predictable results. Science requires that a result be reproducible before it can be accepted as fact, and engineering strives to develop reliable processes for achieving predictable outcomes. Furthermore, there has recently been much discussion of the scientific underpinnings of modeling; Booch, for example, has discussed the notion of modeling-as-science that is worth striving for [11].

In software requirements engineering (RE), ideally, given the same problem domain, the same contexts, and the same good OOA method, independent analysts should produce identical, complete, and high-quality domain models. However, achieving identical, complete domain models is highly unlikely in practice: different analysts may have different abilities and experiences, and these differences often strongly influence the results of performing domain analysis. Moreover, quality is hard to measure, if for no other reason than that different people perceive quality differently. Hence, we settle for the production of measurably *semantically similar* domain models as a sign of the predictability of the results of a method. Therefore, this paper focuses on

the development of a measure of semantic similarity and on evaluating the measure by using it to compare the semantic similarity and predictability of domain models produced for one problem by independent groups of analysts using the two different OOA methods.

In the rest of the paper, Section 2 relates semantic similarity and repeatability. Section 3 describes the research method and related issues. In particular, Section 3.1 describes the background and context of the two case studies, each with a set of domain models produced with the help of one of the OOA methods. Section 3.2 discusses the empirical validity of results. Section 3.3 describes a procedure for comparing the semantic similarity of the domain models in two sets of domain models. Section 4 carries out the actual comparison of the semantic similarity of the elements of each of the two sets of domain models that are in the two case studies. Section 5 describes the results and the lessons learned from the case studies, Section 6 discusses related work, and Section 7 discusses the measures and suggests future work.

2. Semantic Similarity and Repeatability

We use the term *semantic similarity* to indicate the degree of closeness that independently produced domain models of one domain may share. More formally, we use a generalized definition of semantic similarity [12]:

> Semantic similarity, variously also called semantic closeness/proximity/nearness, is a concept whereby a set of documents or terms within term lists are assigned a metric based on the likeness of their meaning/semantic content.

The metric used in this work is domain-expert opinion combined with manual clustering. The actual method used to evaluate semantic similarity is described in Section 3.3 after the presentation of data from the case studies that prompted the discovery of the evaluation method. For now, this section assumes an intuitive notion of semantic similarity.

Semantic similarity is not new and has been used extensively in practice and research. For example, the basis for the notions of reference architectures and design patterns [13], [14] is the idea that all architectures matching a reference architecture and all design patterns matching a particular design are semantically similar.

Repeatability of an OOA method means that given the same domain, different analysts of similar background and experience applying the same OOA method would be expected to produce semantically similar models of the same domain. Thus, semantic similarity of the results of several applications of one OOA method to one domain should be a direct measure of the repeatability of the OOA method.

3. Research Method

The results of this paper come not from any controlled experiment but from an after-the-fact analysis of two distinct sets of university-student-written software requirements specifications (SRSs). The SRSs of the first set were written in one term without the use of UCUM, and the SRSs of the second set were written in a later term using UCUM. After grading the second set, we felt that there was a distinct improvement in the second set over the first. The problems for us were (1) how to measure the change and (2) to determine with this measure if the change amounted to an improvement. The effort to quantify the change led to the semantic similarity measurement procedure described in Section 3.3. During the research to solve these problems, we generated tables, one for each set of SRSs, in which a black cell indicated the presence of an entity in a domain model. A chance glance at these tables side-by-side provided the intuitions that semantic similarity was the measure we sought and that indeed the domain models of the second set were more semantically similar to each other than were the domain models of the first set.

3.1. Description of the Sets of SRSs

The semantic similarity analysis was done on two sets of SRSs for a CBS composed of

1) a Voice-over-IP (VoIP) system and
2) its supporting information management system (IMS).

Production of the specification, in the form of a SRS document, is the term-long project carried out by 3-or-4-person groups in CS445, the first course in a sequence of three software engineering courses that span the final three terms of undergraduate software engineering programs at the University of Waterloo [15]. In later courses, each group designs, implements, tests, and enhances the CBS specified in its SRS.

The project in the three-course sequence involves using various techniques for developing software for real-time systems and OO techniques for developing information systems. Use cases (UCs) [5] are used to capture requirements, and OOA is used as a bridge to later OOD. The real-time components of the CBS are specified using formal finite-state modeling in the Specification and Description Language (SDL) [16]. The information-system components of the CBS are specified using the notations of the Unified Modeling Language (UML) [17]. In addition, students are responsible for modeling user interfaces of the IMS and for the overall management of the requirements specification process.

The semantic similarity analysis was carried out on domain models produced during the writing of the SRSs in two consecutive offerings of the course. Each term's set of domain models formed one of two case studies:

1) CS TOOA[1] consisting of 31 SRSs produced using traditional OOA, and
2) CS UCUM[2] consisting of 34 SRSs produced using UCUM.

The average size of these SRS documents is 120 pages, with actual sizes ranging anywhere from 80 to 250 pages.

3.2. Validity of Results

As already stated, the results of this paper come not from any controlled experiment but from an after-the-fact analysis. Therefore, the analysis lacks the internal validity of a controlled experiment. For a study such as ours, it would be possible to create a controlled experiment only with very small artifacts to ensure sufficient controls and enough subjects. Such a controlled experiment would lack external validity. We believe that the after-the-fact analysis of the SRSs produced by upwards of 30 groups for each treatment in 12-week long RE efforts for a medium-sized VoIP system that approaches industrial size has more external validity than would be possible in any controlled experiment that we can imagine on the same topic.

Another problem with an after-the-fact analysis is that we have no control over what methods the groups actually used in producing their domain models. However, the practical realities of running university courses are that

1) normally, students do not do more than they must to satisfy assignment requirements;
2) normally, students try to do what they must do to produce the documentation they must hand in for grading;
3) since the teaching assistants (TAs) met with the groups regularly, we knew that the groups were behaving normally; and
4) there were assignments throughout each term in which each team was required to hand in specific artifacts, such as a domain model, in their current preliminary forms, for grading and feedback and as a means to ensure that no team waited until the last minute to write the whole SRS in a roughshod way.

Therefore, we are confident that the groups of CS TOOA, which had not been taught UCUM and were not required to make a UC statechart, did not have the information necessary to create a UC statechart. By contrast, the groups of CS UCUM did have this information, as they were taught UCUM and were required to include the UC statecharts they built in the SRSs they handed in.

Whenever the artifacts used in an empirical study are produced by university students instead of by professional requirements analysts, the applicability of any conclusions beyond the university setting can be questioned. On one hand, this study fell out of an attempt to answer why *students* just don't get it when they are taught domain modeling [4]. On the other hand, it would be nice if the results *were* applicable to professional requirements analysts. There is some evidence that upper-division, i.e., third and fourth year, students are representative of junior professionals [18].

Finally, this study is comparing two specific OOA methods, traditional OOA and UCUM, that differ by some steps that do functional analysis in the latter but not in the former. While it would be tempting to generalize the results to all OOA methods that differ by this kind of step, ultimately, the results are about the two specific OOA methods studied. For one thing, building a UC statechart from UCs is not the only way to conduct functional analysis.

3.3. Semantic Similarity Analysis Method

This section describes a generalized manual decision process that given two sets of domain models — call them A and B — determines which set's domain models are more semantically similar to each other.

Suppose that A and B are sets of domain models that have been constructed to model the same system. The comparison is between between two *sets* of domain models, because each set of domain models has been constructed with a method that is different from that used to construct the other. It turns out that it is simpler to ask, "Are the domain models contained in one set more semantically similar to each other than are the domain models contained in the other set?" than it is to ask "In which set are the domain models more semantically similar to each other?". Therefore, the purposes of this procedure are

1) to decide whether the domain models contained in A are more semantically similar to each other than the domain models contained in B, and,
2) if Item 1 is true, to compute the percentage P by which the models in A are more semantically similar to each other than the models in B[3].

This procedure has two phases: a data normalization phase followed by a calculation phase.

The procedure is described as steps to be taken by its performer, whom we call the "data expert", who is called upon to make judgements. For the case studies reported in this paper, the data expert was the first author, whose judgement was based on his experience with the artifacts of the case studies, having been a TA for the course for six years, during which time, essentially the same long-term project was used.

3.3.1. Phase 1: Data Normalization. Domain models that are constructed to model the same problem should have

1. This case study is known as "CS O31VS" in the UCUM paper.
2. This case study is known as "CS N34VS" in the UCUM paper.
3. Clearly, if Item 1's being false makes Item 2 not applicable, then A and B can be swapped to get a true Item 1 and an applicable Item 2.

significant semantic overlap, since they are modeling the same conceptual space, in which a *concept* is anything that has a name in any model. At the same time, they are likely also to exhibit semantic and syntactic variation from each other, since each is created by a different individual or group. The purpose of data normalization is to obtain a normalized view of the concepts in all domain models of one set of domain models. That is, data normalization is to ignore syntactic variation, which is differences in the naming of concepts, in order to be able to show how much the domain models in the set differ, in terms of *semantically unique concepts*. In data normalization, the name of a concept is taken as its value.

To construct a single, representative set of semantically unique concepts in a set D of domain models, first construct the set $RawConcepts(D)$ of all concepts in all of the models in D. Because $RawConcepts(D)$ is a set, *syntactically identical concepts*, i.e., those that have the same name, that appear in more than one domain model appear only once in $RawConcepts(D)$.

Next, partition $RawConcepts(D)$ into equivalence classes of elements that refer to the same semantic concept even if they have different names. This partitioning is done by having the data expert make a number of assumptions:

1) Within a single domain model, the data expert assumes that any name refers to a unique concept, e.g., the use of the word "flight" in one travel agency domain model is assumed to refer to the same concept throughout that domain model.
2) The data expert considers as referring to the same concept, all names, different or the same, from different domain models that refer to what appear to him or her to be the same semantic idea, e.g., "flight", "flightInfo", and "flightNumber" are considered to refer to the same concept if the data expert believes that what they refer to are the same semantic idea. The data expert forms his or her belief by comparing the name, attributes, methods, and relationships of each raw concept with those of all other concepts.

Next, for each equivalence class, chose one member to be the representative concept name, e.g., "flight" for the equivalence class of the previous examples.

Finally, for each concept c_{raw} in each domain model d in D, replace c_{raw} in d with the representative concept name c_{rep} for the equivalence class of $RawConcepts(D)$ to which c_{raw} belongs.

This data normalization process results in a single, unambiguous vocabulary consistent across the set.

There may be errors in the data expert's judgement of semantic equivalence. However, we expect that whatever error there is affects each set about the same so that the comparison can be relied upon to be accurate.

Some question whether a metric based on subjective decisions is any good. When a metric is needed for a concept which is inherently subjective, an exact sum formed of many local, pairwise subjective decisions is is probably more accurate than a single global, overall subjective decision. Such is the underlying principle in COCOMO [19] and in the Analytic Hierarchy Process [20].

The manual process of creating the semantic equivalence classes is labor intensive, slow, and highly subjective. However, it is also likely to be more accurate than any automated approach.

3.3.2. Phase 2: Determination of Semantic Similarity. Given two sets of domain models, A and B, to which data normalization has been applied, the steps for determining if the domain models of A are more semantically similar to each other than are the domain models of B are:

1) Let D be a set of domain models such that $D = A$ or $D = B$.
   In the following, a, b, and d are domain models such that $a \in A$, $b \in B$, and $d \in D$.
2) For any domain model $d \in D$, let $c(d)$ be the set of d's concepts, which are by the data normalization process, d's representative concepts.
3) To get the *major concepts* of any domain model $d \in D$, it will be necessary to prune d to reduce the importance of concepts that appear in only one or a few of the domain models. A concept is said to be k-*significant* if it appears in at least k domain models in D. For some $k \geq 1$, let $mc(d)$ be $c(d)$ restricted to the k-significant concepts. For the case studies reported in this paper, we used $k = 2$, thus ruling out concepts that appear in only one domain model.
   By extension, for a set of domain models D, define the *major concepts* of D,
   $$mc(D) = \bigcup_{d \in D} mc(d).$$
4) Let
   $$cc(A, B) = mc(A) \cap mc(B),$$
   i.e., the set of *concepts* that are *common* to A and B, a.k.a. the *common concepts* of A and B.
   Note that $cc(A, B)$ is *not* the same as the intersection of the concepts of all domain models in $A \cup B$; that intersection would be a much smaller set of concepts. Instead, $cc(A, B)$ contains all of the concepts that belong to at least k domain models in A and at least k domain models in B.
5) For $d \in A \cup B$, let
   $$cc(d) = c(d) \cap cc(A, B),$$
   the set of *common concepts* of A and B that are found in domain model d.
6) Define
   $$cr(d) = \frac{|cc(d)|}{|c(d)|}.$$

This *commonality ratio* measures the number of concepts common to A and B found in a domain model d and measures it relative to d's size. This number is close to 1 if d is very similar to the set of common concepts, and is close to 0 if d is very dissimilar to the set of common concepts, i.e., the number is close to 0 if d contains either few common concepts or many uncommon concepts.

7) Define
$$avg_{cc}(D) = \frac{\sum_{d \in D} |cc(d)|}{|D|},$$
the average number of concepts common to A and B found in each d in D.

8) Define
$$avg_{cr}(D) = \frac{\sum_{d \in D} (cr(d))}{|D|},$$
the *average commonality ratio* relative to A and B of each d in D.

9) Now, it is possible to say that the domain models of A are more *semantically similar* to each other than are the domain models of B if and only if
$$avg_{cc}(A) > avg_{cc}(B)$$
and (*)
$$avg_{cr}(A) > avg_{cr}(B)$$
i.e., if and only if, among A and B, the average number of common concepts found in members of A is higher *and* the *average commonality ratio* of members of A is higher. We chose to use conjunction at the point marked "(*)" above rather than disjunction to be conservative in the judgement of semantic similarity.

10) Finally, if the domain models of A are more semantically similar to each other than are the domain models of B, we define
$$P = avg(avg_{cc}(A)\% - avg_{cc}(B)\%, \\ avg_{cr}(A)\% - avg_{cr}(B)\%),$$
the estimated percentage by which the domain models of A are more semantically similar to each other than are the domain models of B.

It will be necessary to calculate standard errors, standard deviations, and interquartile ranges to get a sense of the distributions of the values.

4. Analysis of Case Studies

This section discusses the semantic similarity analysis of the two case studies, CS TOOA and CS UCUM. Table 1 summarizes the numbers of concepts discovered and captured in all domain models and per domain model of CS TOOA and CS UCUM. The data for the CS UCUM column exclude the data for an outlier domain model with the most semantically unique concepts, while the data for the latter column include the data for this domain model. The excluded domain model has 45 concepts and contains a large number of concepts that we were unable to classify and easily understand what they represent. As can be seen from Table 1, the numbers from each domain model of CS TOOA and CS UCUM end up being very similar.

In particular, the analysis and discussion of the completeness of each domain model is straightforward. The total number of discovered semantically unique concepts is 134 and 110 in CS TOOA and CS UCUM, respectively. The average number of captured concepts in each domain model is 17 and maximum number of captured concepts 33. So, on average, a domain model captured only 12.7% and 15.5% of all valid semantically unique concepts in CS TOOA and CS UCUM, respectively.

Therefore, *no matter what OOA method the groups used*, each domain model they produced was significantly incomplete. The level of incompleteness was so extreme that it was almost not possible to find any correlation between domain models with respect to semantic similarity. To deal with this problem, we ended up constraining the analysis to the 2-significant concepts, those concepts which appeared in at least two domain models of each case study.

4.1. Detailed Evaluation of Concepts in the Domain Models of the Case Studies

The main comparison of the semantic similarity of the concept sets of the domain models in CS TOOA and CS UCUM considers the 36 common concepts that appeared in the domain models of both case studies and in at least two domain models within each case study. Tables 2 and 3 show the distribution of these 36 common concepts in the domain models of CS TOOA and CS UCUM, respectively. The 36 common concepts account for 72% of the major concepts in the domain models in CS TOOA and 77% of the major concepts in the domain models in CS UCUM.

Tables 2 and 3 have the same sideways layout. In each table, each concept c has a row. The row for c is divided into 4 columns, the second of which has subcolumns. The first column contains c's name. The second column is divided into one subcolumn for each domain model d. Row c's entry for the subcolumn for d is black if and only if concept c appears in d. The third column contains the number of domain models that have concept c. This number is the number of black subcolumns in the second column of the same row. The fourth column contains the percentage that the number in the third column is of the total number of domain models.

In the third last row of the table, the subcolumn for any d contains the number of common concepts in d. In the second last row of the table, the subcolumn for any d contains the number of *all* semantically unique concepts in d. In the

Scope	Measure	Case Studies	
		CS TOOA	CS UCUM
In all domain models of the case study	Original, raw concepts	527	622
	Syntactically unique concepts	259	312
	Semantically unique concepts	134	110
In each domain model of the case study	Maximum number of semantically unique concepts	31	33
	Minimum number of semantically unique concepts	8	10
	Average number of semantically unique concepts	17	17
	Median number of semantically unique concepts	16	17

Table 1. CS TOOA – CS UCUM Statistics

last row of the table, the subcolumn for any d contains the commonality ratio for d. Finally, the cell at the intersection of the last row and the last column contains the average commonality ratio for the domain models of the case study, which is the average of all the commonality ratios in the last row.

Tables 2 and 3 show that

1) on average, 84% of all concepts in the domain models of CS UCUM, and
2) on average, 75% of all concepts in the domain models of CS TOOA

are the concepts common to the domain models of both case studies, indicating a higher concentration of common concepts in each domain model of CS UCUM than in each domain model of CS TOOA.

Table 4 shows that the semantic similarity of common concepts in the domain models of CS UCUM is higher than that of CS TOOA. Specifically:

1) The average number of concepts per domain model is approximately 5.5% higher for the domain models of CS UCUM than for the domain models of CS TOOA, as is shown in the cells in Row 2 and Columns 2 and 3.
2) The average number per domain model of common concepts is approximately 14.4% higher for the domain models of CS UCUM than for domain models of CS TOOA, as is shown in the cells in Row 2 and Columns 4 and 5.
3) For each of the average number of concepts per domain model and the average number of common concepts per domain model, the standard error of the average for one case study is approximately the same as the standard error of the average for the other case study, due to the approximately equal size of the two sets of data involved, as is shown in the cells in Row 3.
4) The standard deviation of the average number of concepts per domain model is approximately 35.1% higher for the domain models of CS UCUM than for domain models of CS TOOA, as is shown in the cells in Row 4 and Columns 2 and 3.
5) The standard deviation of the average number of common concepts per domain model is approximately 35.8% lower for the domain models of CS UCUM than for the domain models of CS TOOA, as is shown in the cells in Row 4 and Columns 4 and 5. The probable reason that the standard deviation of the average number of concepts per domain model is higher for the domain models of CS UCUM is the presence in the domain models of CS UCUM of concepts from the outlier DM45. Therefore, we computed also the interquartile range, which can ignore any outlier domain model with an extremely large or an extremely small number of concepts.
6) The interquartile range of the number of concepts per domain model is approximately 47.4% lower for the domain models of CS UCUM than for the domain models of CS TOOA, as is shown in the cells in Row 7 and Columns 2 and 3.
7) The interquartile range of the number of common concepts per domain model is approximately 6.7% lower for the domain models of CS UCUM than for the domain models of CS TOOA, as is shown in the cells in Row 7 and Columns 4 and 5.

That the interquartile range of the number of common concepts per domain model is lower in CS UCUM than in CS TOOA for the same set of data leads to the conclusion that the concept concentration was higher in the domain models of CS UCUM than in the domain models of CS TOOA, for both all and the common concepts. Therefore, the domain models of CS UCUM are more semantically similar to each other than are the domain models of CS TOOA.

Moreover, it is possible to estimate that the semantic similarity of concepts is approximately 10% higher in the domain models of CS UCUM than in the domain models of CS TOOA, based on the previously described quantitative analysis showing that:

- an average increase of 5.5% in the number of concepts per domain model of CS UCUM over the number of

	1	2	3	4	5
1	Metric	CS TOOA Concepts	CS UCUM Concepts	CS TOOA Common Concepts	CS UCUM Common Concepts
2	Average	16.97	17.91	12.29	14.06
3	Standard Error	0.91	1.17	0.63	0.45
4	Standard Deviation	5.07	6.85	3.53	2.60
5	Quartile (.75)	20.00	18.75	14.00	15.75
6	Quartile (.25)	13.00	14.00	10.00	12.00
7	Interquartile Range	7.00	4.75	4.00	3.75

Table 4. CS TOOA–CS UCUM Statistics Summary

concepts per domain model of CS TOOA

- an average increase of 14.4% in the number of common concepts per domain model of CS UCUM over the number of common concepts per domain model of CS TOOA
- an increase in the capture of common concepts per domain model from 75% in the domain models of CS TOOA to 84% in the domain models of CS UCUM; and
- a narrow data spread for both sets of data.

This 10% improvement in semantic similarity came at a cost. We estimate that the cost of teaching UCUM and eliciting requirements for the CS UCUM term was at least about 25% higher than the cost of teaching traditional OOA and eliciting requirements for the CS TOOA term. We do not have data on students' hours, but we do have data on hours the TAs interacted with the students. We assume that the students required more time from the TAs in the CS UCUM term than in the CS TOOA term because the students were spending more time on their project. The first author, as the head TA in both terms, was responsible for answering students' questions found that his workload for the CS UCUM term was about 30% higher than for the CS TOOA term. This time is mostly for teaching, since each student was to go to his or her own project TA for elicitation issues. Also, in each term in CS445, we have each TA report his or her actual workload for the course. The average number of elicitation meetings in a term between a group and its TA, as analysts and customer, increased from about 6–8 per previous non-UCUM-using term to about 10 in the UCUM-using term, i.e., about 25% more. That is, using any variant of UCUM required at least about 25% more elicitation effort.

5. Overall Evaluation and Main Lessons

The first observation is that domain models were very incomplete; i.e., they lacked many needed concepts. The average group using traditional OOA discovered only 12.7% of all valid semantically unique concepts, while the average group using UCUM discovered only 15.5% of the same set of concepts. The maximum percentage of all valid semantically unique concepts in the SRSs of the two case studies found by any group using either method is only 30%. Therefore, we learned the lesson that

> *each OOA method studied produces domain models that lack a majority of the needed concepts.*

Even when using UCUM, which provided analysts with a detailed understanding of a domain's total functionality before they performed conceptual analysis, the resulting domain models were only 15.5% complete. Therefore,

> *detailed understanding of a domain's total functionality does not significantly increase an analyst's ability to discover the concepts in the domain using either studied OOA method.*

Nevertheless, the typical group managed to eventually discover *all* the required functionality that its CBS had to support, even when it did only TOOA with its informal analysis of use cases. However, when a group's domain model was lacking in needed concepts, it tended to assign many of the functions and responsibilities it found to the wrong concepts, namely those that were present in the absence of the correct concepts. Thus,

> *each OOA method studied produces concept-deficient domain models with distorted concept–responsibility assignments.*

Even after constraining the definition of a valid concept from "a semantically unique concept" to "a semantically unique concept *discovered by at least two independent groups of analysts*", the semantic similarity of the domain models of CS UCUM measured only about 10% higher than the semantic similarity of the domain models in CS TOOA. That is, the domain models produced using UCUM are about 10% more predictable than the domain models produced using traditional OOA. The key difference between the methods is that UCUM has an analyst performing a detailed functional analysis before doing conceptual analysis. However, as detailed at the end of Section 4.1, the cost of teaching UCUM and eliciting requirements to novices was about 25% higher for the SRSs of CS UCUM than for the SRSs of CS TOOA. This higher cost would appear to make the small increase in the predictability of the domain models specified using UCUM hard to swallow.

Table 2. CS TOOA Common Concepts

Concepts	CS TOOA Common Domain Model Concepts																													# of DMs	% of DMs
	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15	d16	d17	d18	d19	d20	d21	d22	d23	d24	d25	d26	d27	d28	d29	d30	d31		
untuse-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	30	97
lancal-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	28	90
onepho-40	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	27	87
illbil-41	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	26	84
ordcal-34	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	25	81
csrcsr-42	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	23	74
lersys-28	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	19	61
gercal-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	18	58
gerbil-30	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	15	48
untcsr-34	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	11	35
untpho-32	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	11	35
gerres-29	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	11	35
seruse-41	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	11	35
geruse-25	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	11	35
germai-26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	10	32
berpho-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	10	32
olecon-38	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	9	29
gerhar-29	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	9	29
essip -35	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	8	26
ioddis-30	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	8	26
gerip -35	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	7	23
gespri-35	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	7	23
gercal-25	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	5	16
entpay-38	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	5	16
temsys-39	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	5	16
logtes-37	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	5	16
mertim-40	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	5	16
orderr-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	4	13
gercal-26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	3	10
gercsr-26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	3	10
geraut-23	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
estaut-31	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
iodbil-31	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
gerpri-27	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
aterat-41	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
imetim-41	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
# of Common Concepts	22	19	18	17	17	15	14	14	14	13	13	13	13	12	12	11	11	11	11	11	11	11	10	10	9	9	9	9	8	7	7	Average Commonality Ratio	
# of Concepts	26	24	21	21	20	15	24	22	16	18	15	14	14	18	13	20	19	14	13	12	12	11	17	13	31	16	15	12	20	12	8		
Commonality Ratio	85%	79%	86%	81%	85%	100%	58%	64%	88%	72%	87%	93%	93%	67%	92%	55%	58%	79%	85%	92%	92%	100%	59%	77%	29%	56%	60%	75%	40%	58%	88%	75%	

Table 3. CS UCUM Common Concepts

Concepts	CS UCUM Common Domain Model Concepts																															# of DMs	% of DMs
	d1	d2	d3	d4	d5	d6	d7	d8	d9	d10	d11	d12	d13	d14	d15	d16	d17	d18	d19	d20	d21	d22	d23	d24	d25	d26	d27	d28	d29	d30	d31	d32	d33	d34		
lancal-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	33	97
untuse-33	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	32	94
illbil-41	■	■	■	■	■	■	■	■	■	■	■	■	■			■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	31	91
ordcal-34		■	■	■	■	■	■	■	■	■	■	■			■	■	■	■	■			■	■	■	■	■	■	■	■	■	■	■	■	■	■	28	82
untpho-32	■						■	■	■	■						■	■	■	■	■	■			■			■		■	■	■	■	■	■	■	25	74
untcsr-34	■						■	■	■	■						■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	23	68
geruse-25	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	23	68
gerbil-30	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	22	65
gercal-25	■	■	■	■	■	■	■	■	■		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	22	65
ioddis-30		■					■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	18	53
gespri-35	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	18	53
gerhar-29	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	17	50
csrcsr-42	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	16	47
germai-26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	15	44
essip -35	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	14	41
lersys-28	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	14	41
gercal-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	13	38
gerres-29	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	13	38
gercsr-26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	12	35
entpay-38	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	9	26
gerpri-27	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	9	26
mertim-40	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	9	26
iodbil-31	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	8	24
gercal-26	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	8	24
onepho-40	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	8	24
logtes-37	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	8	24
imetim-41	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	7	21
aterat-41	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	5	15
olecon-38	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	4	12
geraut-23	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
estaut-31	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
orderr-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
gerip -35	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
berpho-33	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
temsys-39	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
seruse-41	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	■	2	6
# of Common Concepts	20	20	19	17	17	16	16	16	16	15	15	15	15	14	14	14	14	14	14	13	13	13	13	13	12	12	12	12	11	11	11	11	10	10	Average Commonality Ratio	
# of Concepts	28	22	21	21	18	23	18	17	16	33	19	18	17	45	25	17	17	16	14	18	14	13	13	13	15	14	13	12	14	14	12	11	18	10		
Commonality Ratio	71%	91%	90%	81%	94%	70%	89%	94%	100%	45%	79%	83%	88%	31%	56%	82%	82%	88%	100%	72%	93%	100%	100%	100%	80%	86%	92%	100%	79%	79%	92%	100%	56%	100%	84%	

On the other hand, the two numbers, 10% and 25%, are not comparable and should not be used to make a direct cost–benefit analysis of performing a detailed functional analysis of the entire domain before doing conceptual analysis. For example, one possible benefit of spending the 25% more time eliciting requirements is more detailed requirements and UCs. As described in the UCUM paper, the TAs and the first author agreed qualitatively that the requirements and UCs specified in the SRSs of the UCUM-using terms were more detailed and consistent than those of the non-UCUM-using terms. It is necessary also to take the entire lifecycle into account. It is hard, if not impossible, to estimate the total benefits to the downstream development speed and to the reliability, robustness, and other qualities of the CBS being developed caused by the 10% improvement in predictability of the domain models and by the use of UCUM. It is possible also that the benefits of using UCUM increase as an analyst becomes a more experienced UCUM practitioner.

6. Related Work and Counter Indications

The UCUM paper describes related work by Glinz; Whittle and Schumann; and Harel, Kugler, and Pnueli [8]–[10] whose qualitative conclusions generally agree with those of our work.

The results of the current paper showing the effectiveness of UCUM as an approach in which an analyst does detailed functional analysis *before* doing conceptual analysis contradicts, at least superficially, the conclusions of a case study by Kabeli and Shoval [21]. Their case study shows that doing data modeling before doing functional analysis leads to better OO models, as judged by their criteria [21], than doing functional analysis before doing data modeling. It is not surprising that early case studies produce results that appear to contradict each other. First of all, neither our nor their case study is conclusive, and neither traditional OOA nor UCUM perform explicit *data* analysis, while the subjects of the Kabeli and Shoval study did explicit *data* analysis. Second, there may be yet other parameters, entirely overlooked in each case study, that consistently account for the contradictory conclusions. Only additional, independent, experimentation in the future can resolve this issue.

7. Conclusion

This paper suggests semantic similarity as a measure for estimating the predictability of the domain models produced by OOA methods. It exercises the measure in a comparison of the predictablity of the models produced by two specific OOA methods, (1) traditional OOA and (2) UCUM, as a representative of OOA methods that encourage detailed functional analysis prior to conceptual analysis. The paper compares the degree of semantic similarity and completeness among the sets of domain models for the same problem produced by the two methods. The comparison is based on two after-the-fact case studies:

- Traditional OOA was carried out to produce 31 SRSs of a large VoIP CBS containing an IMS. Data from these 31 SRSs and their domain models were gathered later in the case study CS TOOA.
- UCUM was carried out to produce 34 SRSs of the same large VoIP CBS containing an IMS. Data from these 34 SRSs and their domain models were gathered later in case study CS UCUM.

One conclusion of the two case studies is that *the use of UCUM yields domain models that are about 10% more semantically similar to each other than are the domain models yielded by traditional OOA.*

The second conclusion of the two case studies is that despite the 10% difference in the semantic similarity of the models produced by the two OOA methods, *each OOA method produces domain models that lack a majority of the needed concepts.*

More work needs to be done to test the conclusions as hypotheses in controlled experiments and to validate semantic similarity of artifacts as a measure of the predictability of methods to produce these artifacts.

Acknowledgment

We thank all students and TAs who were in the classes in which UCUM was developed and applied; we thank in particular those who provided feedback. We thank also Ric Holt, John Mylopoulos, and Ladan Tahvildari, the independent reviewers of this research work for their comments and suggestions.

Davor Svetinovic's work was supported in part by Canadian NSERC Postgraduate Scholarship PGS B-255929-2002 and a Canadian FCAR Doctoral Scholarship. Daniel Berry's, Michael Godfrey's, and Nancy Day's work was supported in part by Canadian NSERC Grant Numbers NSERC-RGPIN227055-00, NSERC-RGPIN217236-99, and NSERC-RGPIN240537-01, respectively.

References

[1] R. Matulevičius and P. Heymans, "Comparing goal modeling languages: An experiment," in *Requirements Engineering: Foundation for Software Quality (REFSQ)*, ser. LNCS 4542, P. Sawyer, B. Paech, and P. Heymans, Eds. Springer, 2007, pp. 18–2007.

[2] A. Davis, O. Dieste, A. Hickey, N. Juristo, and A. M. Moreno, "Effectiveness of requirements elicitation techniques: Empirical results derived from a systematic review," *Proceedings of the IEEE International Requirements Engineering Conference (RE)*, pp. 179–188, 2006.

[3] P. Merrick and P. Barrow, "Testing the predictive ability of a requirements pattern language," *Requirements Engineering Journal*, vol. 10, no. 2, pp. 85–94, 2005.

[4] D. Svetinovic, D. M. Berry, and M. Godfrey, "Concept identification in object-oriented domain analysis: Why some students just don't get it," in *Proceedings of the IEEE International Conference on Requirements Engineering RE'05*, 2005, pp. 189–198.

[5] C. Larman, *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and the Unified Process*, 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall, 2001.

[6] D. Svetinovic, D. M. Berry, N. A. Day, and M. W. Godfrey, "Unified use case statecharts: Case studies," *Requirements Engineering Journal*, vol. 12, no. 4, pp. 245–264, 2007.

[7] D. Harel, "Statecharts: A visual formalism for complex systems," *Science of Computer Programming*, vol. 8, no. 3, pp. 231–274, 1987.

[8] M. Glinz, "An integrated formal model of scenarios based on statecharts," in *Proceedings of the 5th European Software Engineering Conference*, 1995, pp. 254–271.

[9] J. Whittle and J. Schumann, "Generating statechart designs from scenarios," in *ICSE '00: Proceedings of the 22nd International Conference on Software Engineering*, 2000, pp. 314–323.

[10] D. Harel, H. Kugler, and A. Pnueli, "Synthesis revisited: Generating statechart models from scenario-based requirements," in *Lecture Notes in Computer Science*, ser. LCNS, vol. 3393, 2005, pp. 309–324.

[11] G. Booch, *Object-Oriented Analysis and Design with Applications*, 2nd ed. Redwood City, CA, USA: Benjamin/Cummings, 1994.

[12] "Semantic similarity definition," http://en.wikipedia.org/wiki/Semantic_similarity; accessed September 15, 2006.

[13] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design Patterns*, 1st ed. Hoboken, NJ, USA: John Wiley & Sons, 1996.

[14] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, and M. Stal, *Pattern-Oriented Software Architecture: A System of Patterns*, 1st ed. Boston, MA, USA: Addison-Wesley, 1995.

[15] "SE463/CS445 course project," http://www.student.cs.uwaterloo.ca/~cs445/; accessed January 30, 2006.

[16] R. Bræk and O. Haugen, *Engineering real time systems: an object-oriented methodology using SDL*. London, UK: Prentice-Hall International, 1993.

[17] J. Rumbaugh, I. Jacobson, and G. Booch, *The Unified Modeling Language Reference Manual*, 2nd ed. Reading, MA, USA: Addison-Wesley, 2004.

[18] P. Berander, "Using students as subjects in requirements prioritization," in *ISESE '04: Proceedings of the 2004 International Symposium on Empirical Software Engineering*, 2004, pp. 167–176.

[19] B. W. Boehm, *Software Engineering Economics*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1981.

[20] T. L. Saaty, *The Analytic Hierarchy Process*. New York, NY, USA: McGraw Hill, 1980.

[21] J. Kabeli and P. Shoval, "Data modeling or functional analysis: What comes next? an experimental comparison using FOOM methodology," in *Procedings of the Eighth CAISE–IFIP WG 8.1 International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'03)*, 2003, pp. 48–57.