Requirements Eng (2004) 9: 238–247
DOI 10.1007/s00766-004-0205-5

ORIGINAL ARTICLE

Joonhee Yoo · Joseph Catanio · Ravi Paul
Michael Bieber

Relationship analysis in requirements engineering

Received: 15 July 2003 / Accepted: 25 May 2004 / Published online: 7 October 2004

Abstract This research addresses a major shortcoming in today's requirements analysis techniques—the lack of a rigorous and comprehensive process to explicitly capture the relationship structure of the problem domain. Whereas other analysis techniques lightly address the relationship discovery process, *relationship analysis* (RA) is a systematic, domain-independent analysis technique focusing exclusively on a domain's relationship structure. This paper describes RA's taxonomy of relationship types and corresponding brainstorming questions for eliciting the relationship structure from a domain expert. A preliminary case study analysis of online bookstores using RA as well as a formal experiment have both confirmed RA's effectiveness in helping the analyst produce significantly higher quality requirements. RA should become an invaluable tool for analysts, irrespective of the software engineering approach taken during systems analysis.

J. Yoo
Graduate School of Management, Rutgers University,
University Heights, Newark,
NJ 07102, USA

J. Catanio
Mathematics and Computer Science Department,
College of Arts and Sciences, La Salle University,
Philadelphia, PA 19141, USA
E-mail: catanio@lasalle.edu

R. Paul
Department of Decision Sciences, College of Business,
East Carolina University, Greenville,
NC 27858, USA
E-mail: paulr@mail.ecu.edu
URL: http://www.business.ecu.edu/users/paulr

M. Bieber (✉)
Information Systems Department,
College of Computing Sciences,
New Jersey Institute of Technology,
University Heights, Newark, NJ 07102, USA
E-mail: bieber@oak.njit.edu
URL: http://web.njit.edu/~bieber

Keywords Relationship analysis · Requirements analysis · Requirements engineering · Entity-relationship · Taxonomy · Brainstorming · Communications · Software engineering · Systems analysis and design

1 Motivation

Ambiguous, missing, or incomplete requirements contribute to the high failure rate of systems development projects [1]. This usually results from incomplete requirements gathering or inaccurate communication between the analysis and design phases. Improving these two aspects of the requirements engineering process thus is crucial to the overall improvement of software development project success.

Requirements are not fully collected, in part due to the lack of a formal process or structure to assist the analyst in eliciting all the available information. The introduction and increasing use of the use-case model has provided some much needed help. Unfortunately, use-cases only provide information about the "actors" in the system and the steps they undertake to perform a function. They do not explicitly address identifying the collection of relationships between the entities (or objects) in the system.

There has been significant progress in the last few years towards solidifying the "engineering" in software engineering by providing repeatable and standardized methods such as the unified process (UP) and the unified modeling language (UML). Using the standardized use-case model notations and templates, an analyst could elicit and document the requirements to carry out the function of interest in a very comprehensive manner. This would allow the analyst to identify the objects and to some extent the collaboration between these objects, and communicate these to the designer via the use-case diagram. The next step in the UP is for the designer to generate the class diagram [similar to the entity-rela-

tionship (ER) diagram of the structured analysis (SA) methodology]. However, a conceptual and procedural gap between these two critical steps fails to take into account some of the following issues: How does the analyst know that all the relationships among the objects have been identified? How does he or she know that the different ways in which the objects are related have been identified? And finally, how could the analyst best communicate the collection of relationships discovered during the elicitation stage to the designer in the most complete and effective way?

We developed the *relationship analysis* (RA) technique and associated support tools described in this paper to address two crucial shortcomings of the requirements phase—relationship identification and relationship communication. Our goal is to help the analyst in completing a comprehensive requirements gathering process, especially regarding relationship information, and communicating this rich information accurately to the designers in a formal diagram.

In Sect. 2, we discuss the notion of multiple relationships forming the context around an entity or object, and further motivate our research. Section 3 contrasts RA with several other analysis techniques. Section 4 describes the RA approach and Sect. 5 provides a short case study. Section 6 gives an overview of an experiment we conducted, showing that RA produces richer analyses than an object-oriented analysis (OOA) technique. Section 7 concludes with a general discussion, including RA's contributions and boundaries, and some future research directions.

2 Relationships

A domain's interrelationships constitute a large part of its implicit structure. A deep understanding of the domain relies on knowing how all the entities or objects are interconnected. Although commonly used, the relationship construct is poorly defined and lacks a strong theoretical foundation [2, 3]. Relationships are a key component of vital design artifacts such as ER diagrams and object-class diagrams. These diagrams capture an important, but often rather limited subset of relationships, leaving much of the domain's relationship structure out of the design and thus out of the model of the

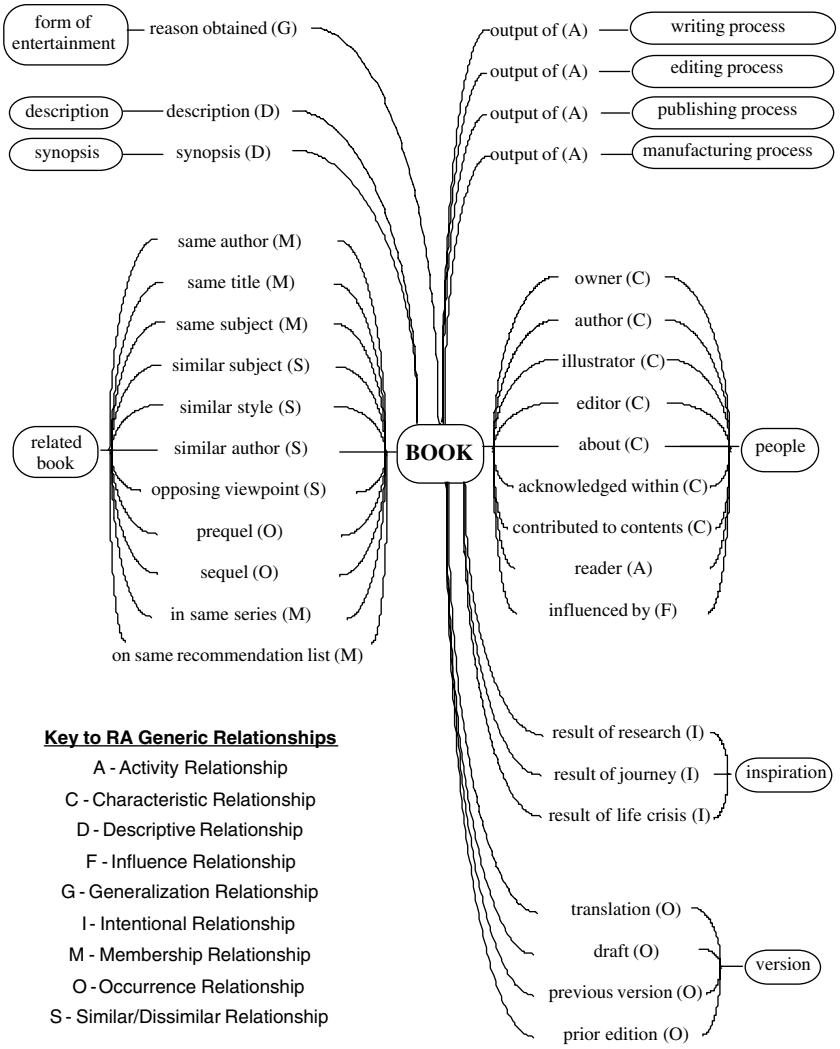

Fig. 1 A subset of the relationships around books found through the relationship questions in Table 2, which were based on Table 1's relationship taxonomy

system. While analyses and models are meant to be focused, and, thus, limited representations of a system, we believe that the incomplete relationship specification is not by design, but rather from the lack of any technique to determine them explicitly. Many analyses thus miss aspects of the systems they represent, and often do not convey all the useful information they could when passed on to the designers. It seems that formally and rigorously identifying a complete set of relationships early in the development process has not been a primary concern of software engineers.

A rich plethora of relationships surround many objects in the real world [4]. For example, a product may have several relationships to its customers, who can purchase it, recommend it to others, provide input for modifying it, make comments on it, transform it for other uses, dispose of it, trade it for other goods, etc. Often, a typical analysis would only capture the first of these. Figure 1 presents a subset of the relationships around a book, which one may wish to include, e.g., in a library support application. (The full set would be at least half again as large [5].) Note the presence of multiple relationships between pairs of objects.

Yet, the literature indicates that object-oriented models classify relationships into three primary categories, namely generalization, aggregation, and association [6–19]. Surely there must exist more than three categories of relationships exist! (Indeed, as we note in Sect. 4.1, many more do.)

Once identified in the analysis, the most effective subset of these relationships could be included in the design and then implemented as links. When relevant, designers could specify that several links be displayed for a given object.

So, how does one go about *discovering the relationships* among objects/classes? And once discovered, how does one *communicate the relationships* to the designer in a formal manner? RA specifically addresses these concerns and offers solutions that fill a vital gap in systems analysis.

Relationship analysis provides systems analysts with a systematic technique for determining the relationship structure of an application, helping them to discover all potentially useful relationships in application domains and to document them effectively.

Relationship analysis enhances users', analysts' and system developers' understanding of application domains by broadening and deepening their conceptual model of the domain. Developers can then enhance their implementations by including additional links and other representations of the relationships.

Relationship analysis can be used either to thoroughly describe an existing application (or information domain) in terms of its relationships, or as part of a systems analysis to understand a new application being designed. It provides a comprehensive technique to perform a systematic analysis for identifying and modeling relationships in a generic domain.

3 An overview of life cycle methodologies

The analysis phase of the systems development life cycle strives to precisely and comprehensively isolate and understand the problem domain, and document what is to be built. Software engineering has several established methodologies to support the activities during this phase. In this section, we briefly highlight how RA could support (supplement) the analysis phase of the two most common life cycle methodologies—SA and OOA.

Structured analysis (SA) Structured analysis (SA) is the most popular approach to problem analysis. Although, SA is *process* and *data* oriented, its primary focus is to determine what data needs to be transformed by the system while maintaining a degree of separation between the process and the data. SA uses functional decomposition to map from problem domains to functions and subfunctions. Because of the emphasis on data, SA extensively makes use of analysis tools such as data flow diagrams, which do not capture relationships, and ER diagrams, which capture merely a subset of relationships. SA techniques at best provide general guidelines for discovering relationships as opposed to encompassing a systematic approach.

The ER diagram deals primarily with entities, their attributes and relationships [20]. An ER diagram maps from the real world into entities, attributes and relationships. It provides a way to express problem domain understanding by direct mapping. ER diagrams for the most part allow only a single relationship to connect two entities. Also, the analysis techniques for developing ER diagrams provide, at most, ad hoc approaches for determining the relationships. Often it is assumed that the relationships are obvious between any two entities, and that an analyst will see them intuitively.

Object-oriented analysis (OOA) While SA is still widely used, especially in the United States, OOA is rapidly gaining popularity around the world. OOA was developed by combining the concepts of semantic data modeling and object-oriented programming languages. OOA methodologies focus on *objects* and recommend the modeling of object classes including their attributes and behaviors as well as their relationships through the mechanism of message passing.

OOA uses popular tools such as use-cases and class diagrams extensively to document the processes and objects, and make it easier to move from the analysis stage to design and then onto development. It provides, however, at most an ad hoc approach to documenting relationships, the focus being more on objects and their interactions via messages. We believe that the systematic nature of RA makes it an accessible approach to analysts of varying experience levels.

None of the existing methodologies explicitly help analysts in determining the detailed relationship struc-

ture of the application domain, and therefore they are not as comprehensive as analysts treat them. For any analysis methodology to be truly effective, it needs to be systematic, controlled and comprehensive. RA is a systematic, controlled technique that can supplement and "complete" the existing approaches.

4 Relationship analysis

Relationship analysis is a relationship elicitation process based in a thorough taxonomy of the relationships found in a computer application's domain [5, 21, 22]. Each of the taxonomy's 16 categories has a series of exploratory questions, which an analyst asks a domain expert or user in order to elicit an implicit set of relationships in his or her mind. This section presents both the taxonomy and some brainstorming questions. In Sect. 6, we describe an experiment showing that RA yields a richer design than a corresponding OOA.

4.1 RA's generic relationship taxonomy

Table 1 presents RA's generic, domain-independent relationship taxonomy.

These relationship categories were developed on the basis of a very extensive literature review [21] and trial-and-adjustment prototyping. Yoo [21] compares RA's taxonomy with ten other domain-specific taxonomies in detail, with additional comparisons with over 20 others. RA's categories encompass all of these other taxonomies' relationships. This includes, for example, OOA [13] (which provides RA's generalization/specialization, whole-part, classification/instantiation and association relationship classifications).

Generalization/specialization relationships concern the relationships among objects in a taxonomy [23–25]. Self-relationships include characteristic, descriptive and occurrence relationships.

Table 1 RA's 16 generic relationships

1. Generalization/specialization	
Self	2. Characteristic
	3. Descriptive
	4. Occurrence
Whole-part/composition	5. Configuration/aggregation
	6.Membership/grouping
7. Classification/instantiation	
Comparison	8. Equivalence
	9. Similar/dissimilar
Association/dependency	10. Ordering
	11. Activity
	12. Influence
	13. Intentional
	14. Socio-organizational
	15. Temporal
	16. Spatial

Whole-part/composition relationships include configuration/aggregation relationships [26, 27] and membership/grouping relationships [18, 19, 28]. Classification relationships connect an item of interest and its class or its instance.

Comparison relationships break down into similar/dissimilar and equivalence relationships, involving such relationships as in thesaurus or information retrieval [29, 30]. Association/dependency relationships break down into ordering, activity, influence, intentional, socio-organizational, spatial and temporal relationships. The term association and dependency could be used interchangeably, because every association involves some concept of dependency [18, 19]. Because association is defined as a relationship that is defined by users, there could be no fixed taxonomy for it. The association relationship taxonomy is fluid compared with other relationships. The current association relationship taxonomy is based on our observations, analyses, ontologies [31], and existing classifications [18, 19].

Ordering relationships involve some kind of sequence among items. Activity relationships are created by combining SADT activity diagrams [31] and case relationships [32] to deal with relationships associated with activities or actions abstractly. This relationship could cover any activities that involve input or output, and deal with agents and objects involved in the activities. Influence relationships exist when one item has some power over the other items. Intentional and socio-organizational relationships could be identified in intentional (meaning/opinion) and social ontologies, respectively. Temporal [33, 34] and spatial [35–37] relationships deal with these perspectives.

Each relationship category can be further broken down into lower levels of detail, from which we derived a basic set of brainstorming questions. Yoo [21] details each lower-level category and the literature from which we derived each.

4.2 Conducting a relationship analysis

Relationship analysis begins with a detailed use-case analysis. From the use-cases we identify stakeholders and the entities or "items of interest" that will form the anchors for relationships. For each item of interest identified by the domain expert or user, the analyst asks a series of questions to elicit the relationships around it, which actually often leads to discovering additional elements of interest that these connect.

Table 2 gives a series of brainstorming questions that an analyst uses to elicit domain information from the user. Each set of questions is derived from the lower levels of detail for each relationship in the taxonomy, described in [21]. For the purposes of this paper, the questions in Table 2 are rather condensed and highly generic. They should be tailored to each item of interest. For example, the descriptive rela-

Table 2 Sample brainstorming questions emanating from RA's generic relationships

Generalization/specialization	Is there a broader term for this item of interest? Is there a narrower term for this item of interest?
Characteristic	What attributes and parameters does this item of interest have?
Descriptive	Does this item of interest have a description, definition, explanation, or a set of instructions or illustrations available within or external to the system?
Occurrence	Where else does this item of interest appear in the application domain? What are all the uses of this item of interest?
Configuration/aggregation	Which components comprise this item? What materials are used to make this item? What is it a part of? What phases are in this whole activity?
Membership/grouping	Is this item a segment of a larger item? Is this item a member of a collection? Are these items dependent on each other in a group?
Classification/instantiation	Is this item of interest an example of a certain class? If a class, which instances exist for this element's class?
Equivalence	What is this item of interest equal or equivalent to in this domain?
Similar/dissimilar	Which other items are similar to this item of interest? Which others are opposite to it? What serves the same purposes as this item of interest?
Ordering	What prerequisites or preconditions exist for this item? What logically follows this item for a given user's purpose?
Activity	What are this item's inputs and outputs? What resources and mechanisms are required to execute this item?
Influence	What items (e.g., people) cause this item to be created, changed, or deleted? What items have control over this item?
Intentional	Which goals, issues and arguments involve this item of interest? What are the positions and statements on it?
	What are the comments and opinions on this item? What is the rationale for this decision?
Socio-organizational	What kinds of alliances are formed associated with this item of interest? Who is committed to it in the organizational structure?
	Who communicates with it or about it, under what authority and in which role?
Temporal	Does this item of interest occur before other items? Does this item occur while other items occur?
Spatial	Which items is this item of interest close to? Is this item of interest nearer to destination than other items? Does this item overlap with other items?

tionship prompts analysts to ask whether an item of interest has a definition, explanation, set of instructions or illustrations available within or external to the system. (These are all lower-level categories for the generic relationship "descriptive.") The analyst clearly should ask each of the questions individually, and in a way that makes the most sense to the particular domain expert.

5 Case study

We recently applied RA to the domain of on-line bookstores. Some of the relationships we found have already been implemented in bookstore Web sites, but many do not appear there. The analyst could conduct a follow-on cost/benefit implementation analysis, which would show that many are not cost effective to provide and others might give users access to competitors, which an e-commerce Web site will often avoid. Yet, some are useful. And several of the others that the RA implementation analysis of a bookstore would reject, provide opportunities for third parties to sell, or libraries and other governmental services to provide their services to benefit the common good. In any event, we were amazed at the scope of the relationships we found that do not appear on the Web, yet seem so obvious once we performed the RA. (Anecdotally, we found that analysts and users returning to redesign applications were awed at how much knowledge RA elicited, which was missing within existing system implementations [38].)

The following is a selection of the relationships discovered for the element "book." Figure 1 includes many of these, as well as others from a fuller analysis, which can be found in Yoo and Bieber [5].

5.1 Generalization/specialization

Using the *specialization* relationship, we determined that a book is an abstraction of the objects novel and short story. Often, customers prefer collections of short stories or full novels.

Using the *generalization* relationship, we realized that books could serve several different roles. For example, books could be generalized into "reading materials," and that many other kinds of reading materials exist besides books that an on-line bookstore could provide. Books are also a kind of product, and an on-line bookstore could consider other kinds of products such as videos. These roles vary based on the customer's intent (looking for something quick to read, looking for something for a long trip, looking for a gift, looking for something to amuse me this evening), and an on-line bookstore could expand to serve several of these intents.

5.2 Characteristic relationship

Using the generic *characteristic* relationship led us to the following characteristics of books in which different customers might be interested:

- Relevance (How long will this book be relevant? For example, a road map or set of statistics might be valid for a month, a year or a decade.)
- Owner (Who owns the copyright on this work?)
- Contributors (authors, illustrators, editors, people interviewed during its authoring)
- Intent (reference, history, how to, self help, tutorial, etc.)
- Awards received
- Ratings (from different consumer groups)

5.3 Occurrence

Both customers and systems analysts will be interested in various occurrence relationships for books:

- Where is this book listed? (bestseller lists)
- Where has this book been reviewed or discussed?
- Are there translations available?
- Are there newer/older/early/draft versions available, perhaps under a different name?
- Does this book have prequels or sequels?
- Who (else) sells this book?

The occurrence relationship also leads to warnings an on-line bookstore could provide:

- "You already have a copy of this book in your shopping basket. Are you sure you want another?"
- "You purchased this book last week, but it has not been delivered yet."
- "You purchased this book last December."

5.4 Configuration/aggregation

Using the domain independent categories for the generic *configuration/aggregation* relationship, we determined that a book is related to the following objects that it contains:

- Its chapters (Is the table of contents available? Can the customer read the first chapter?)
- Its index (giving an indication of the book's level of detail and expertise)
- Its foreword (which might entice a customer)
- Its introduction (giving an indication of the book's level of detail and expertise)
- Its illustrations (e.g., customers may be enticed by the illustrations in a children's book or figures in a technical book).

Using the *configuration/aggregation* relationship, we recalled that a book may also be a part of a series. The customer may wish to see other books in the series.

5.5 Activity

The generic *activity* relationship leads us to ask who and what uses books, and how:

- Which kinds of people read a certain book (Which types of customers might want to buy this book?)
- People give books as gifts (Is the bookstore's Web site set up to facilitate people looking for gifts?)
- Book groups (Is the bookstore doing anything to support book groups?)

Using the generic *activity* relationship we also determined which objects are inputs to a book:

- Paper (Is the paper of good quality? Is it printed on recycled paper?)
- Binding (Was the book manufactured to last?)
- Cover (Is it hardcover or softcover?)

The generic *activity* relationship also prompts us to ask which activities a book results from:

- Result of research
- Result of a journey
- Result of a crisis in the author's life

5.6 Influence

The generic *influence* relationship leads us to ask which people, events, philosophies, and other books might have influenced the author or the subject matter of a book. Customers fascinated by a book (or author) might want to learn more about these influences.

6 Experiment

We conducted an experiment to compare RA with other systems analysis techniques. OOA by Coad and Yourdon [12] was used as the traditional OOA method. The subjects were 96 undergraduate students enrolled in four sections of a software engineering course. Each section served as one group: one control group, one using RA, one using OOA, and one using both techniques in conjunction with each other. After a training session, the subjects were asked to identify the objects and relationships for an on-line bookstore.

This task was concerned with identifying modeling entities and relationships, and had nothing to do with how to represent them (a future research topic). In each section, subjects were allowed to represent their analysis using any representation scheme including simple lists.

6.1 Measures

The first dependent variable measured was the number of modeling entities plus the number of relationships

identified in the analysis task. The data on this variable were collected by counting the number of modeling entities and relationships specified in the analysis results, removing the duplicates in the count. This is an indicator of the analytical power of the analysis methodology used. A higher value of this dependent variable would indicate deeper analysis or understanding of the application domain.

The second dependent variable is the quality of the analysis. Four expert judges graded the analysis results based on quality. Each expert judge graded all analysis results. Two criteria were used to evaluate the quality of analysis results. The first was whether the analysis results were relevant for the problem domain and task. The second criterion whether the analysis included important modeling entities and relationships in the problem domain.

The teaching notes and questionnaires are available from the authors.

6.2 Quantity of analysis results

We defined problem domain understanding as the number of different modeling entities plus the number of different kinds of relationships identified (when duplicates are eliminated). Table 3 indicates the average number of different modeling entities plus the number of different kinds of relationships identified. The numbers in the parentheses for the *No Methodology* and *OOA* conditions indicate objects, attributes, and relationships identified, respectively. Objects here include anything other than attributes and relationships. The numbers in parentheses for the *RA* and *OOA + RA* conditions indicate elements and relationships identified, respectively. Elements here include anything other than relationships and are equivalent to modeling entities.

Table 3 shows that the group that used RA has the highest average among the four options. Also, these results show little difference between the groups that used RA and both OOA and RA. A big difference exists between the number of kinds of relationships identified by the RA and OOA groups. More relationships identified enabled identification of more modeling entities.

Table 3 Average number of different modeling entities plus the number of different kinds of relationships identified by experimental subjects in the experiment

	Not using RA	Using RA
Not using OOA	No methodology (objects, attributes, relationships) **24.4** (11.1, 10.9, 2.5)	RA (modeling entities, relationships) **66.7** (53.7, 13.1)
Using OOA	OOA **32.9** (14.3, 15.9, 2.7)	OOA + RA **65.4** (54.1, 11.3)

Numbers were rounded to the nearest tenth. The control group (no methodology) was assigned no analysis technique. The three treatment groups used; RA, OOA, and both RA and OOA in conjunction with each other

Table 4 ANOVA for the number of different modeling entities plus different relationships identified

Source	Sum of squares	df	Mean square	*F*	Significance
No methodology	31,920.8	3	10,640.3	36.8	0.00
OOA	281.0	1	281.0	1.0	0.33
RA	31,071.1	1	31,071.1	107.5	0.00
OOA + RA	529.6	1	529.1	1.8	0.18

All numbers are rounded to the nearest tenth, with significance rounded to the nearest 100th.

Table 5 *t*-Tests for pairs of the four experimental conditions

	t	df	Significance (two-tailed)
RA vs. OOA	−6.4	51	0.000
RA vs. OOA +RA	−0.2	34	0.869
OOA + RA vs. OOA	−6.5	47	0.000
No methodology vs. OOA	−2.9	58	0.005
No methodology vs. OOA + RA	−8.9	41	0.000
No methodology vs. RA	−8.2	45	0.000

t-Test figures are rounded to the nearest tenth

Table 4 presents an analysis of variance (ANOVA) for the number of different modeling entities plus different relationships identified. It shows that RA has a main effect. OOA has no main effect due to the size of RA's main effect when compared with that of OOA when both RA and OOA are used. Using both OOA + RA apparently does not provide any advantage in discovering modeling entities and relationships over just using RA.

Table 5 presents the *t*-tests for each pair of four conditions. RA has a clear advantage over OOA. There is little difference between RA and OOA + RA. OOA + RA is better than OOA. All three options (RA, OOA, and OOA + RA) have a higher average than No Methodology. (Note that OOA has a clear advantage over No Methodology when RA is not involved.).

6.3 Quality of analysis results

We now turn to the quality of the analysis, as scored by our team of expert judges. Table 6 presents the results, graded on a scale from 1 (lowest) to 7 (highest). It shows that the RA group had a higher average quality analysis than the No Methodology or OOA groups. The judges found very little quality difference between the RA and OOA + RA groups.

The empirical results confirm that RA is better than no methodology or a representative OOA in terms of the

Table 6 Average quality of the analyses in each experimental group, graded on a scale from 1 (lowest) to 7 (highest)

	Not using RA	Using RA
Not using OOA	3.35 (no methodology)	5.21 (RA)
Using OOA	3.11 (OOA)	5.23 (OOA + RA)

Numbers were rounded to the nearest 100th

quantity and quality of modeling entities and relationships identified. The results show the potential of RA to supplement system analysis methodologies, confirmed that RA helps system analysts with a deeper problem domain understanding, and to identify useful relationships in the application domains.

Further experimental results [21] confirm that RA is better than no methodology or the representative OOA in terms of usability. This result shows the practical nature of RA.

7 Discussion

In this section, we discuss RA's contributions and boundaries, as well as some future research.

7.1 Contributions

This research addresses a major shortcoming in today's analysis techniques—the lack of a formal process to identify relationships in a system being modeled. To the best of our knowledge, RA is a systematic, domain-independent analysis technique focusing exclusively on a domain's relationship structure.

Relationship analysis serves two major purposes. First, it helps users, analysts and designers develop a deeper understanding of the application domain (through making the relationships explicit). Second, RA should result in fuller and richer application analyses and designs.

Relationship analysis will have a positive impact on the design of the systems that people use everyday. It should become an invaluable tool in the toolkit of the analyst irrespective of the software engineering approach taken during the analysis. RA could very easily become a standard extension to the other tools and techniques currently available for analysis. We are designing RA to fit into the UP and into UML modeling. RA should result in richer Web sites that give deeper and broader access to information. RA also should result in higher quality software applications, both on and off the Web.

7.2 Clarifications and future research

Relationship analysis combines components from many well-known classifications into a comprehensive relationship taxonomy. During RA brainstorming, people sometimes have the impression that the categories overlap. This impression can arise for several reasons. First, the domain expert does not always make the distinction between one category's relationships and another's. While working on one category, they may residually think of a relationship from a former category. Second, recall that two entities can be connected by multiple relationships. For example, from the focus of the membership relationship, two books may be related in that they belong to the same series. From the focus of the ordering relationship, these same books may be related in that one is a sequel to the other. From the focus of the similarity relationship they may be related in that they deal with the same subject. While brainstorming about one relationship category, our minds will naturally focus on the entities as well as the current relationship category and we may think of some of the other relationships once we discover the first, even if the others formally fall under different categories. Finally, the relationships themselves are inherently interrelated, causing us to naturally associate relationships from different categories [21]. These interrelationships in fact enhance the RA process; subconsciously people will continuously review prior relationship categories and probe new ones throughout the brainstorming session, thereby eliciting a much fuller representation of the relationship structure.

As such, although RA may be able to characterize systems thoroughly, it is not possible to claim it categorically complete, since it is not based on a theoretical model. Wand et al. [39] supports the idea that theories related to human knowledge can be used as foundations for modeling in systems development. As part of our future research endeavors, we are currently developing an RA model, based on theory, to categorize relationships. Our intent is to use concepts from ontology, concept theory, classification theory, and speech act theory to develop the model.

For clarification, we wish to highlight some aspects that RA is not, or that our current research does not address. RA is not a design technique. Rather it is a method-independent analysis, which provides useful input to the systems design phase. Future research will investigate effective integration of RA diagrams such as Fig. 1 into design documents such as the class diagram. We also shall investigate automatic generation of design documents from the analysis documentation. Furthermore, being method-independent, RA's systematic approach to discovering a domain's relationship structure could complement many different systems analysis approaches. As part of future research we shall investigate formally integrating RA with other methodologies.

Relationship analysis does not provide algorithms to generate relationships. RA is strictly an elicitation technique embodied in a systematic procedure. A systematic process is an essential element to process improvement [40]. A systematic approach to knowledge elicitation makes requirements gathering and problem

understanding less dependent on the experience level of the process engineer [41]. A systematic approach to requirements elicitation helps to improve accuracy and provide a greater level of detail.

We intend RA to provide a high degree of support to the analyst and *not* to replace the analyst by totally automating the relationship discovery and documentation process. There can be no substitute to the quality and expertise provided by the human analyst. However, we believe that RA can significantly enhance the effectiveness of the human analyst. To this end we are currently designing an RA template for analysts to complete, which will feed in to the RA diagram of Fig. 1. We also plan to develop computerized software to assist the analyst in these procedures.

We realize that not all relationship types will apply to every application, and that analysts might not have time for a full RA analysis. Thus we plan to customize and scale RA to different time availabilities and to serve different purposes. As part of the scaling and customization, we need to determine which relationship types are most likely to apply to which general kinds of elements of interest, for which general kinds of domains, and for which general kinds of users. In the RA software, we would customize the RA Templates, presenting just a subset of possible relationships for a given kind of element for the analyst to consider.

8 Conclusions

The RA process presented in this paper addresses a significant need in the requirements elicitation and analysis process. None of the popular analysis and design methodologies such as the SA or the OOA and design nor the SDLC methods such as the UP provide a formalized approach to elicit and document a domain's relationships. Further, while the UML provides the class diagram to document the objects and their relationships once identified, it does not include a diagram that assists the analyst in identifying the relationships and then communicating the relationships to the designer.

We envision RA seamlessly integrating into the requirements elicitation and analysis process irrespective of the analysis methodology used. As RA, like the UML, is methodology-independent, it can be equally effective in development efforts using the structured approach or the object-oriented approach or even a hybrid approach. Further, with the use and eventual incorporation of the RA diagram into the UML, analysts and designers will have a standard and formal diagram to communicate relationships information to each other.

An analyst collecting information about the functionality of the system using use-cases could also use RA to gather relationship-specific information. These two approaches can and should be used iteratively to gather the wealth of information needed to build and deploy successful systems. Just as use-cases could identify the primary entities (or objects) to get RA started, the information pertaining to relationships elicited during the RA process can be used effectively to identify new use-cases. Together, these two approaches when used during the critical requirements phase can provide a depth of information never before available to analysts, designers and developers. Preliminary analyses of bookstores using RA, as well as a formal experiment, have both confirmed that RA is not only easy to use but also helps the analyst produce significantly higher quality output, especially in the number of relationships.

In conclusion, RA will significantly enhance the systems analyst's effectiveness, especially in the area of relationship discovery and documentation, which will ultimately result in the development of higher quality software applications.

Acknowledgements The authors thank Ashish Ghoda, Il Im, Robb Klashner, and Atanu Pal for their assistance with this RA research. We gratefully appreciate partial funding support for this research by the United Parcel Service, the New Jersey Center for Pervasive Information Technology, the New Jersey Commission on Science and Technology, and the National Science Foundation under grants IIS-0135531 and DUE-0226075.

References

1. The Standish Group (1995) Chaos, home page at http://www.standishgroup.com/chaos_resources/index.php
2. Siau K (1997) Theoretical foundations for relationship construct in information modeling—relation element theory. In: Proceedings of the 19th annual conference of the association for information systems (AIS'97), Indianapolis, Indiana, August 1997, pp 622-624
3. Siau K (1996) Empirical studies in information modeling: interpretation of the object relationship. PhD dissertation, University of British Columbia, Canada
4. Catanio J, Nnadi N, Zhang L, Bieber M, Galnares R (2004) Ubiquitous metainformation and the WYWWYWI (what you want, when you want it) principle. J Digital Inform (JoDI) 5(1)
5. Yoo J, Bieber M (2000) A relationship-based analysis. In: Hypertext 2000 proceedings, ACM Press, San Antonio
6. Kobryn C (2000) Modeling components and frameworks with UML. Commun ACM 43(1):31–38
7. Booch G, Rumbaugh J, Jacobson I (1999) The unified modeling language user guide. Addison-Wesley, Reading
8. Booch G (1986) Object-oriented development. IEEE Trans Softw Eng 12(2):211–221
9. Rumbaugh J, Blaha M, Premerlani W, Eddy F, Lorensen W (1991) Object-oriented modeling and design. Prentice Hall, Englewood Cliffs
10. Shlaer S, Mellor S (1992) Object life-cycles: modeling the world in states. Prentice-Hall, Englewood Cliffs
11. Ross DT (1977) Structured analysis: a language for communicating ideas. IEEE Trans Softw Eng 3(1):16–34
12. Coad P, Yourdon E (1991) Object-oriented analysis. Yourdon Press, Englewood Cliffs, New Jersey
13. Martin J, Odell J (1995) Object-oriented methods: a foundation. Prentice Hall, Englewood Cliffs
14. Jacobson I, Christerson M, Jonsson P, Overgaard G (1992) Object-oriented software engineering. A use case driven approach. Addison-Wesley, Reading
15. Embley D, Kurtz B, Woodfield S (1992) Object-oriented systems analysis: a model-driven approach. Prentice-Hall, Englewood Cliffs

16. De Champeaux D, Faure P (1992) A comparative study of object-oriented analysis methods. J Object Oriented Programming March/April:21–33
17. Firesmith D (1993) Object-oriented requirements analysis and logical design: a software engineering approach, Wiley, New York
18. Henderson-Sellers B (1998) OPEN relationships—associations, mappings, dependencies, and uses. J Object Oriented Programming November/December:51–72
19. Henderson-Sellers B (1997) OPEN relationships—compositions and containments. J Object Oriented Programming November/December:51–72
20. Chen P (1976) The entity-relationship model—toward a unified view of data. ACM Trans Database Syst 1(1):9–36
21. Yoo J (2000) Relationship analysis. PhD Dissertation, Rutgers University
22. Yoo J, Bieber M (2000a) Towards a relationship navigation analysis. In: Proceedings of the 33rd Hawaii international conference on system sciences. IEEE Press, Washington DC
23. Borgida A, Mylopoulos J, Wong H (1984) Generalization/specialization as a basis for software specification. In: On conceptual modeling: perspectives from artificial intelligence, databases, and programming languages. Springer, Berlin, Heidelberg New York, pp 87–117
24. Brachman R (1983) What IS-A is and isn't: an analysis of taxonomic links in semantic networks. IEEE Comput October:30–36
25. Smith J, Smith D (1977) Database abstractions: aggregation and generalization. ACM Trans Database Syst 2(2):105–133
26. Brodie M (1981) Association: a database abstraction for semantic modelling. In: Chen PP (ed) Entity-relationship approach to information modeling and analysis, proceedings of the 2nd international conference on entity-relationship approach (ER'81), Washington, DC, October 1981. ER Institute, pp 583–608
27. Motschnig-Pitrik R, Storey V (1995) Modelling of set membership: the notion and the issues. Data Knowl Eng 16:147–185
28. Odell J (1994) Six different kinds of composition. J Object Oriented Program January:10–15
29. Belkin N, Croft W (1987) Retrieval techniques. Annu Rev Inf Sci Technol (ARIST) 22:109–131
30. Neelameghan A, Maitra R (1978) Non-hierarchical associative relationships among concepts: identification and typology. Part A of FID/CR report no. 18. FID/CR Secretariat Document Research and Training Center, Bangalore
31. Mylopoulos J (1998) Information modeling in the time of the revolution. Inf Syst 23(3/4):127–155
32. Fillmore CJ (1968) The case for cases. In: Universals in linguistic theory. Holt, New York
33. Allen J (1983) Maintaining knowledge about temporal intervals. Commun ACM 26(11):832–843
34. Frank A (1998) Different types of times in GIS. In: Egenhofer M, Golledge R (eds) Spatial and temporal reasoning in geographic information systems, chap 3, pp 41–62
35. Cobb M, Petry F (1998) Modeling spatial relationships within a fuzzy framework. J Am Soc Inf Sci 49(3):253–266
36. Egenhofer M, Herring J (1990) Categorizing binary topological relations between regions, lines, and points in geographic databases. Technical Report, Department of Surveying Engineering, University of Maine
37. Rodriguez M, Egenhofer M, Rugg R (1999) Assessing semantic similarities among geospatial feature class definitions. In: Vckovski A (ed) Interop '99, Zurich, Switzerland. Lecture Notes in Computer Science, vol 1580. Springer, Berlin Heidelberg New York
38. Bieber M (2001) Supplementing applications with hypermedia. Technical Report, IS Department, NJIT
39. Wand Y, Monarchi D, Parsons J, Woo CC (1995) Theoretical foundations for conceptual modeling in information systems development. Decis Support Syst 15:285–304
40. Becker-Kornstaedt U (2001) Towards systematic knowledge elicitation for descriptive software process modeling. In: Proceedings of the PROFES, pp 1–18
41. Bandinelli S (1995) Modeling and improving an industrial software process. IEEE Trans Softw Eng 21(5):440–454