A Systems Approach to Product Line Requirements Reuse

Nan Niu, *Member, IEEE*, Juha Savolainen, Zhendong Niu, Mingzhou Jin, and Jing-Ru C. Cheng

***Abstract*—Product line engineering has become the main method for achieving systematic software reuse. Embracing requirements in a product line's asset base enhances the effectiveness of reuse as engineers can work on the abstractions closer to the domain's initial concepts. Conventional proactive approaches to product line engineering cause excessive overhead when codifying the assets. In this paper, we propose a systems-oriented approach to extracting functional requirements profiles. The validated extraction constructs are amenable to semantic case analysis and orthogonal variability modeling, so as to uncover the variation structure and constraints. To evaluate our approach, we present an experiment to quantify the extraction overhead and effectiveness and a case study to assess our approach's usefulness. The results show that our automatic support offers an order-of-magnitude saving over the manual extraction effort without significantly compromising quality and that our approach receives a positive adoption rate by systems engineers.**

***Index Terms*—Product line engineering, requirements engineering, reuse in systems engineering, software reuse.**

I. INTRODUCTION

IN today's market, engineers are pressured to quickly deliver high-quality systems and systems of systems that provide increasingly ambitious functionality. Meanwhile, they can no longer afford, in terms of time or money, to build every system from scratch. Product line engineering aims to ameliorate this situation by codifying a reusable core asset base so that individual systems can be developed in a prescribed and economical way [1], [2].

Adopting product line engineering in practice is not without risk. Conventional methods follow the *proactive* model, i.e., making an upfront investment to develop reusable assets for reuse and deriving products by using the assets [3]. Parnas aptly summarized the dilemma faced in proactive product line development: We had to design the core assets for a product family at a time when we could not possibly know what members of the family would actually be built [4]. To resolve such a paradox, Krueger proposed the *extractive* adoption model as a means of reusing existing products for the product line's initial baseline [5]. The extractive approach is particularly effective for an organization that has accumulated development experiences and artifacts and wants to quickly transition from conventional to product line engineering [3].

Although much of the research to date has focused on code reuse, embracing requirements in the asset base has many advantages [2], [6]–[9]. Not only was reuse identified early, but the effectiveness of reuse was enhanced as engineers can work on the abstractions closer to the system's initial concepts [10]. Contemporary proactive approaches to developing product line requirements require experts to perform heavyweight domain analysis [11] that make knowledge acquisition difficult to automate and extend. Our research is aimed at providing automated support for easily *extracting* reusable requirements with *lightweight* techniques.

In particular, we propose a systems-oriented approach based on information retrieval (IR) techniques to automatically identify functional requirements profiles (FRPs) by analyzing natural language (NL) documents as the overwhelming majority of requirements are written in NL [12]. We adopt the orthogonal variability model (OVM) [13] to represent the extraction result and then use Fillmore's case theory [14] to characterize each FRP's semantics and form an initial product line OVM.

The contributions of our work lie in the concept of FRP. Our approach complements existing domain analysis methods by quickly offering insights into system functionalities and variabilities, and the approach is scalable and extensible. To mitigate the risk of being overgeneral, domain concepts are incorporated when possible. We evaluate our approach on two product lines: automarkers and traffic management systems. The results show that our automatic support offers an order-of-magnitude saving over the manual extraction effort without significantly compromising quality, and our approach receives a positive adoption rate by potential users.

The remainder of this paper is organized as follows. Section II situates the extractive model within the strategies to develop product line assets and reviews related work. Section III articulates the extraction and modeling of FRPs. Section IV describes an experiment to quantify the cost-effectiveness of FRP extraction. Section V presents a case study to assess the scope of applicability and usefulness of our approach. Section VI draws some concluding remarks and outlines future work.

Manuscript received November 23, 2009; revised March 8, 2013; accepted April 10, 2013. Date of publication June 21, 2013; date of current version August 21, 2014. This research is supported in part by the U.S. Army Engineer Research and Development Center under Grant W912HZ-10-C-0101, by the U.S. Department of Transportation under Grant DTOS59-09-G-00058, and by the U.S. National Science Foundation under Grant CCF-1238336.

N. Niu is with the Department of Computer Science and Engineering, Mississippi State University, MS 39762 USA (e-mail: niu@cse.msstate.edu).

J. Savolainen is with the Danfoss Power Electronics A/S, 6300 Graasten, Denmark (e-mail: JuhaErik.Savolainen@danfoss.com).

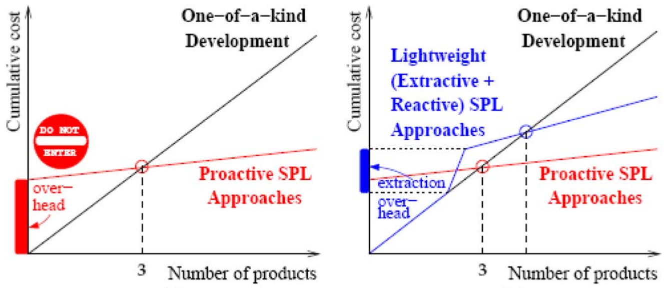
Z. Niu is with the School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China (e-mail: zniu@bit.edu.cn).

M. Jin is with the Department of Industrial and Information Engineering, The University of Tennessee, Knoxville, TN 37996 USA (e-mail: jin@utk.edu).

J.-R. C. Cheng is with the Information Technology Laboratory, U.S. Army Engineer Research and Development Center, Vicksburg, MS 39180 USA (e-mail: ruth.c.cheng@usace.army.mil).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/JSYST.2013.2260092



(a) (b)

Fig. 1. ROI schemas. (a) Being proactive often causes excessive overhead. (b) Being extractive and reactive can reduce the overhead.

II. BACKGROUND AND RELATED WORK

A. *Strategies to Develop Product Line Assets*

Implementing a reuse program in a corporate environment requires a decision concerning when and where capital investment is to be made. Product line practitioners have followed different strategies to develop the core assets [3].

1) With the *proactive* model, an organization makes an upfront investment to develop the core assets for reuse so that the products can be developed with reuse. Although this approach can be effective in a mature and stable domain, it demands a large upfront investment and sometimes causes an abrupt transition from an organization's existing practice [5].
2) In the *reactive* model, reusable assets are developed as needed when reuse opportunities arise. This approach does not require much upfront effort and may work well in a volatile domain, but the cost for reengineering and retrofitting existing products with reusable assets can be high without a well-thought-out architecture [3].
3) The *extractive* model reuses one or more existing software products for building the product line's initial asset base. This approach can be effective for an organization that has accumulated development experiences and artifacts in a domain and wants to quickly transition from conventional one-of-a-kind software development to product line engineering [3].

The return on investment (ROI), among other factors, is crucial to an organization's product line adoption. Fig. 1 sketches the ROI curves [15]; note that no specific scale is defined on the axes since the schemas are provided merely for illustration. In one-of-a-kind development without any reuse, the cumulative cost increases in a linear fashion. In proactive approaches, as shown in Fig. 1(a), a certain amount of upfront effort is needed to develop the core assets. This investment pays off later in that each product is simply a tailoring of the core assets. Therefore, the more the products derived from the family, the more the savings that we will have, compared to the no-reuse scenario. According to the rule of thumb most often found in the literature [1], the break-even point in Fig. 1(a) is 3, i.e., one will see the benefit after building the third product. This is a very good number because it is hard to imagine a product line without at least three family members.

However, a practical concern is the excessive overhead shown in Fig. 1(a). As Parnas pointed out [4], assets themselves are not sellable, so this overhead often makes engineers feel that there is much risk involved and much effort wasted, particularly when no product has yet been developed. It is like putting up a "DO NOT ENTER" sign to block the practitioners from taking the path to reach the break-even point or beyond.

To reduce the upfront effort, Krueger introduces lightweight approaches to first extract the assets from existing products and then reactively accommodate the changes [5]. Fig. 1(b) depicts the ROI of such a model. It is only after several products are developed that one starts codifying the core assets. Since one does not have to be perfect for the first time when extracting the assets, more effort is needed, compared to proactive approaches, to coevolve the assets and the products. The break-even point of lightweight approaches may be delayed, as indicated in Fig. 1(b). In order to reach to the break-even point, the extraction overhead must be as low as possible.

B. *Identifying and Modeling Product Line Requirements*

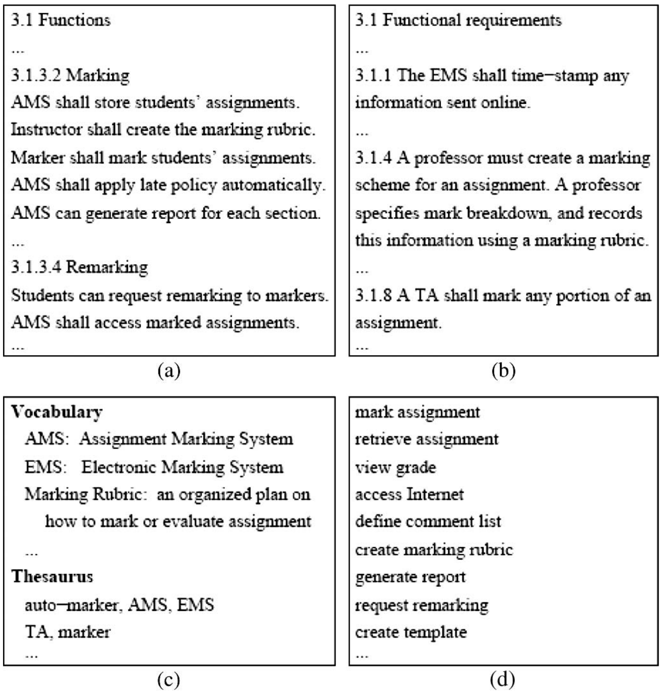
Domain analysis has been the predominant way of defining a product line's requirements assets [16]. One of the drawbacks refers to its intrinsic domain dependence. Domain analysis methods count on experts' experience and intuition to manually acquire domain knowledge. Namely, there are no rules that enable systems engineers to identify domain elements easily [11]. We aim to complement domain analysis via lightweight and automated techniques.

Much product line research has focused on modeling functional requirements because system functionality represents the very noticeable aspect of a feature, which is an identifiable abstraction of an application domain that must be implemented, tested, delivered, and maintained [17]. While system qualities, such as reusability and sustainability, may become the prominent features in the long run, functionalities remain the product line's salient features directly observable by users, customers, and other stakeholders.

Halmans and Pohl extend use cases to model the product line's essential variability from a system usage perspective. Essential variability includes functionality, system environment, data, etc. The authors explicate two concepts in use cases, the variation point and variant. The purpose is to support an intuitive representation of customer-relevant variability aspects [18].

Moon *et al.* [11] elicit domain use cases by means of primitive requirements, which represent complex requirements with an exact paraphrase consisting of simpler words. Sample primitive requirements in the e-forum domain are "write an opinion" and "search a scrapbook," which, like use cases, are expressed in verb–direct object (DO) pairs.

Liaskos *et al.* [19] identify variability in requirements goal models by carefully examining the semantic characterization of every goal's OR-decompositions. They refine the high-level goal into a set of verb–DO tasks, such as "send message" and "display record." Their work also illuminates the importance of distinguishing between intentional variability and background variability.



3.1 Functions
...
3.1.3.2 Marking
AMS shall store students' assignments.
Instructor shall create the marking rubric.
Marker shall mark students' assignments.
AMS shall apply late policy automatically.
AMS can generate report for each section.
...
3.1.3.4 Remarking
Students can request remarking to markers.
AMS shall access marked assignments.
...

(a)

3.1 Functional requirements
...
3.1.1 The EMS shall time–stamp any information sent online.
...
3.1.4 A professor must create a marking scheme for an assignment. A professor specifies mark breakdown, and records this information using a marking rubric.
...
3.1.8 A TA shall mark any portion of an assignment.
...

(b)

Vocabulary
AMS: Assignment Marking System
EMS: Electronic Marking System
Marking Rubric: an organized plan on how to mark or evaluate assignment
...
Thesaurus
auto–marker, AMS, EMS
TA, marker
...

(c)

mark assignment
retrieve assignment
view grade
access Internet
define comment list
create marking rubric
generate report
request remarking
create template
...

(d)

Fig. 2. Scenario for extracting requirements profiles. (a) SRS for Assignment Marking System. (b) SRS for Electronic Marking System. (c) Domain concepts. (d) FRPs.

Our work on FRPs reported here makes the verb–DO linguistic clue explicit and operational. As Bosch points out, starting from the product line's functional requirements should not preclude the optimization of quality requirements during the architectural design stages [20], [21]. In fact, we have explored the use of quality attribute scenarios to study requirements modularity and interactions [22].

III. FRPs

A. Motivating Example

We motivate our work with a set of Web-based automarker systems, which were designed for automating the process of marking first-year programming assignments. Twelve teams, each consisting of three to four junior undergraduates, conducted requirements analysis and wrote software requirements specifications (SRSs) for their course projects [23]. All 12 automarker SRSs followed the IEEE-830 standard in a textual form [24]. Fig. 2(a) and (b) shows the excerpts from two SRSs in the repository.

We are interested in culling a set of FRPs from these SRSs. We define FRPs to be the action-oriented concerns [25] that bear a high information value of a document [26]. FRPs model the user-visible system functionalities and are represented by "verb–DO" pairs. Fig. 2(d) shows a partial list of FRPs extracted from the automarker SRSs.

Product line engineering considers it crucial to define a set of standard terms used in discussions about and descriptions of the domain [16]. Fig. 2(c) depicts a snippet of the domain concepts: Thesaurus identifies synonym classes [27], whereas vocabulary provides the definitions of terms, acronyms, and abbreviations required to properly interpret the requirements documents [24]. These concepts are identified by domain experts. According to Fig. 2(c), we would treat "marking rubric" as a single conceptual unit and thus determine the FRP "create marking rubric," as indicated in Fig. 2(d).

B. Extracting FRPs

The central question that we address in this section is the following: Given an NL document, how can its characterizing attributes, which relate to system functionalities, be produced? When constructing the indices for a requirements artifact, IR [27] techniques draw information from the texts rather than from a human expert. Automatic indexing systems attempt to characterize the document rather than understand it. We prefer IR techniques in our work for reasons of cost, scalability, and domain transportability [26].

Extracting valuable conceptual information from documentation can be done by using rich indexing units. Maarek *et al.* used a two-word unit, called lexical affinity (LA), for profiling software libraries [26]. LAs in large textual corpora have been shown to convey information on both syntactic and semantic levels and to provide us with a powerful way of taking context into account.

For our purposes, we restrict the definition of LAs by observing them within a finite requirements document rather than within the whole language so as to retrieve *conceptual* affinities rather than purely lexical ones. One limitation of considering only a two-word unit as an LA is that domain concepts are not preserved. For example, "marking rubric" would be treated as two separate words, not as one proper term, in [26]. To address this problem, we augment our approach with a semantic component, as shown in Fig. 2(c), so that each entry in the domain vocabulary is maintained as one atomic conceptual unit.

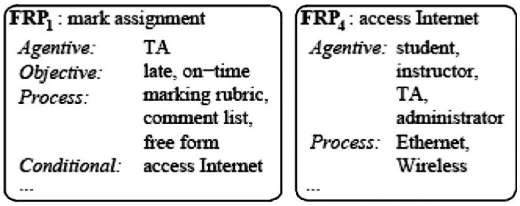
We have developed an IR-based algorithm for extracting domain-aware LAs [23]. The main idea is to make use of an empirical observation that 98% of lexical relations relate words which are separated by, at most, five words within a single sentence [28]. The window is slid throughout the document without crossing sentence boundaries. Given that the window size and the domain vocabulary entries are bounded by some small constants, the extraction of domain-aware LAs is linear in the number of conceptual units in the document [23].

We define the information value ρ of the extracted LA based on both the LA's frequency of appearance in the text and the quantity of information of the conceptual units involved [23]. The LAs with high ρ scores thus effectively characterize the requirements document, but they typically include several modifier-modified relations. Consider the following sentence taken from the Electronic Marking System SRS in Fig. 1(b):

> "A professor specifies mark breakdown, and records this information using a marking rubric."

Some of the potential LAs in this sentence are:

- of type verb–DO, e.g., "specify breakdown" and "record information";
- of type subject–verb, e.g., "professor specify"; or
- of type noun–noun, e.g., "mark breakdown."



FRP_1 : mark assignment
Agentive: TA
Objective: late, on–time
Process: marking rubric, comment list, free form
Conditional: access Internet
...

FRP_4 : access Internet
Agentive: student, instructor, TA, administrator
Process: Ethernet, Wireless
...

Fig. 3. Semantic cases for FRPs.

We are concerned only with the verb–DO relation since our goal is to construct functional profiles. Shepherd studied the verb–DO pairs in source code and observed their denotations of action-oriented concerns [25]. More generally, an especially strong relationship exists between verbs and their themes in English. A theme is the subject matter that the action (implied by the verb) acts upon and usually appears as a DO. Thus, we define the FRPs of a document to be the domain-aware LAs that have a high information value (ρ) and bear a verb–DO relation.

C. Modeling FRPs

Although the extracted and validated FRPs are capable of characterizing the product line's action themes, the flat list [e.g., Fig. 1(d)] hinders us from gaining insights into the variability structures and dependences. We use Fillmore's case theory [14] as a basis for understanding language semantics in a systems engineering context, although, here, we focus on functional requirements. The theory analyzes the surface syntactic structure of sentences by studying the combination of *cases* (i.e., semantic roles like agent, object, location, etc.) which are required by a specific verb. Fig. 3 shows two sample case structures: Each case defines a variation dimension for the FRP, and a case's values determine the range of that dimension. For example, only a "TA" can "mark assignment," and the types of assignment to be marked can be "late" or "on-time."

According to Fillmore, there exists an essential set of cases that fits in the case system of every known language. Each of these universal cases addresses a particular semantic concern of the verb in a sentence, and each represents a potential semantic slot that may or must be associated with the verb. FRPs have made the DO role explicit because the verb–DO relation renders the action and its theme in English [28]. The discovery of variation structures can be driven by identifying the essential cases associated with the verb in every FRP. In this context, a case defines a variation dimension, i.e., a question whose alternative answers result in alternative refinements of the original action-oriented concern expressed by the FRP. The collection of all dimensions relevant to an FRP determines the *variation structure*, or the variation frame, evoked by the FRP.

Following Fillmore's idea of defining a universal set of cases, we introduce a general set of dimensions for conceptualizing the FRP's variation structure.

1) *Agentive* defines the agent(s) whose activities will bring about the FRP's state of affairs. Responses to this question are typically (combinations of) actors found in the domain, including the system-to-be. For example, $\{\text{machine}, \text{TA}, \text{instructor}\}_{\text{Agentive}}$ "check time stamp."

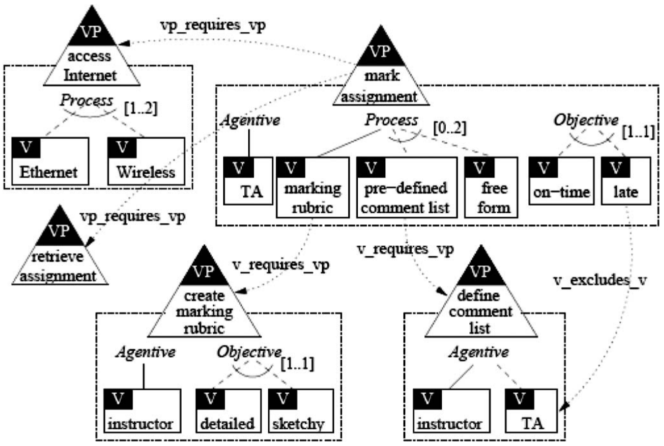


Fig. 4. Partial OVM for the automarker product line.

2) *Objective* defines the object(s) that is affected by the FRP's activity. Since a DO is already part of the FRP, this case concerns mainly with the *types* of DO involved. For example, "mark $\{\text{late}, \text{on-time}\}_{\text{Objective}}$ assignment."
3) *Locational* defines the spatial location(s) where the FRP's activity is supposed to take place. For example, "mark assignment" $\{\text{in the lab}, \text{at home}\}_{\text{Locational}}$.
4) *Temporal* defines the duration or frequency of the FRP's activity. For example, "keep log" for $\{\text{a term}, \text{a month}, \text{a week}\}_{\text{Temporal}}$.
5) *Process* refers to the instrument used, as well as the means and the manner by which the FRP's activity is performed. Some examples are, "access Internet" via $\{\text{Ethernet}, \text{Wireless}\}_{\text{Process}}$, "mark assignment" $\{\text{following marking rubric}, \text{in free form}\}_{\text{Process}}$, or "adjust mark" $\{\text{dramatically}, \text{subtly}\}_{\text{Process}}$.
6) *Conditional* defines the trigger(s) of the FRP's action or the condition(s) under which the FRP's function can be achieved. For example, "mark assignment" only if $\{\text{"access Internet"}, \text{"retrieve assignment"}\}_{\text{Conditional}}$.

The set is by no means an exhaustive list of grammatical features that must be associated with functional requirements descriptions but a catalog of categories that can help analysts understand the variation points, i.e., *what* can vary, of the FRP. Our experience showed that systematically identifying the variation point could uncover its variants (*how* it varies) that would otherwise remain hidden. For instance, it was when "mark *late* assignment" was identified that we noticed that *on-time* assignments should be marked as well.

We take advantage of the OVM notations [13] to rigorously express the product line's variability. Fig. 4 illustrates an OVM, in which a "VP" triangle represents a variation point (what can vary) and a "V" box represents a variant (how the "VP" varies). A mandatory variant is linked by a solid line, whereas optionals are linked by dotted lines. The alternative choice among the optionals is further annotated with an arch, along with the cardinalities specified in $[min \ldots max]$. The variability constraints are given by arrows in Fig. 4.

To map semantics to OVM, we treat FRPs as variation points since FRPs capture the domain's action-oriented concerns and every product in the product line should address these concerns

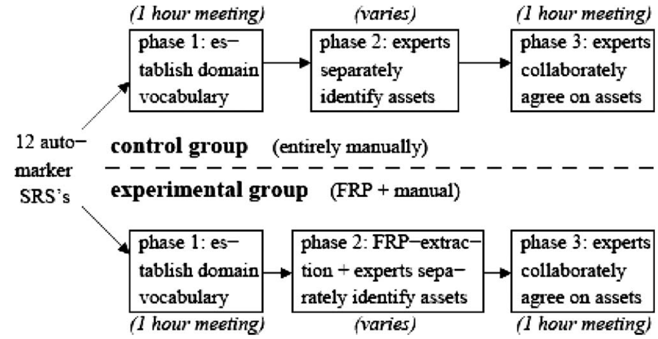


Fig. 5. Experimental design.

in one way or another. We now discuss the intra- and inter-FRP variability issues [13]. Our purpose is to identify the variability dependences and constraints so that FRPs can be integrated to form the product line's asset base. To that end, we have proposed several heuristic rules [23] for variability interdependence identification. It is important to keep in mind that variability management requires a deep understanding of the domain. Our heuristics serve as an aid to this understanding and should be treated as such. Our work is guided by the OVM framework [13]. As shown in Fig. 4, we extend the OVM by adding a boundary for each FRP to mark its internal variation structure and organizing the variants by the FRP's semantic cases. The idea is to allow the user to zoom in (display) or zoom out (hide) the internal structure of any FRP to gain a comprehensive view of the OVM.

IV. EXPERIMENTAL EVALUATION

We designed a quasi-experiment to assess FRP extraction's cost-effectiveness. We compared manual extraction with FRP-based extraction. Our goal was to quantify the extraction overhead [cf. Fig. 1(b)] and the amount of support that FRP extraction could provide. It is important to keep in mind that FRP extraction is not a replacement, but a complement, to existing domain analysis methods. Its purpose is to reduce the manual operation cost so that the analyst can make the best use of his/her time and expertise.

A. Setting

We used the 12 automarker SRSs in our experiment (cf. Section III-A). The average automarker SRS was about 40 pages long, containing about 800 sentences. Fig. 5 shows our experimental design. The task was to extract functional requirements assets from the 12 automarker SRSs. Following family-oriented abstraction, specification, and translation [16], one of the most mature product line methods, we divided the task into three phases: The first was to establish domain vocabulary, the second was to ask the experts to separately identify the assets, and the third was to have experts collaboratively discuss their findings and agree on an asset base. We recruited six domain experts: The control group comprised one instructor and two teaching assistants (TAs); the experimental group comprised one instructor, one TA, and one student. Allocation was not randomized but was based on the time availability of the individual experts.

Our independent variable was the asset extraction method: The control group performed the task entirely manually; the experimental group followed the "FRP-first, manual-second" method. We fully implemented the FRP-extraction procedure described in Section III-B. In comparison, we also generated verb-based single-term indices, ranked by the frequency of occurrence (Verb_Freq) and the information value (Verb_INFO). The FRPs are ranked by the ρ scores.

Our dependent variables were cost and effectiveness. We measured cost in terms of expert-hour. As shown in Fig. 5, both groups spent the same time in phases 1 and 3, so the extraction effort spent in phase 2 would contribute to the cost difference. We measured effectiveness using well-known IR metrics, precision and recall [27]. We also assessed the extent to which experts agreed when comparing their extraction results.

B. Results

We identified 15 acronyms and 6 synonym classes from the SRSs in phase 1 and made such domain vocabulary available to both groups prior to phase 2. It took each individual in the control group about 10 h to manually extract the functional requirements assets. Therefore, $\text{extraction-cost}_{\text{Manual}}$ was about 30 expert-hours.

The experimental group was supported with extracted FRPs. It took our FRP-extraction implementation less than 30 min to process the 12 SRSs. Averagely speaking, 17 significant FRPs were extracted from each SRS. Each expert then spent approximately 50 min to validate the FRPs and produce the assets. Therefore, $\text{extraction-cost}_{\text{FRP+Manual}}$ was about 3 expert-hours, which was one order of magnitude less than $\text{extraction-cost}_{\text{Manual}}$.

To assess the effectiveness of FRP extraction, we measured precision and recall, in comparison with the control group's manual extraction results. The effectiveness comparison was performed by measuring, for the indexing schemes, precision at several levels of recall. In particular, three steps were involved [27]. First, for each document, compute precision at fixed recall values; this is achieved by looking at the top elements from the retrieval results so that the varying recall values can meet the pre-fixed values. Second, for each given recall value, compute the average precision over all the documents in the data set. Finally, connect the precision averages to extrapolate the entire curve.

We have built such curves for Verb_Freq, Verb_INFO, and FRPs. The curves are shown in Fig. 6 where ten fixed recall values are plotted. The best performance is reached by the scheme whose curve is closest to the upper right corner of the graph. The bump of the FRP curve is due to the inability of four SRSs to reach the 30% recall level or beyond; for the remaining eight SRSs, the average precisions keep decreasing for the recall values greater than 30%. The Verb_Freq curve slightly indicates such a fluctuation. The Verb_INFO curve, to our surprise, is so flat that the indices are indifferent. This may suggest that Verb_INFO should not be applied directly, but it certainly warrants further investigation.

The results in Fig. 6 show that, for the sample SRSs, the FRPs are better characterizations of system functionalities than the

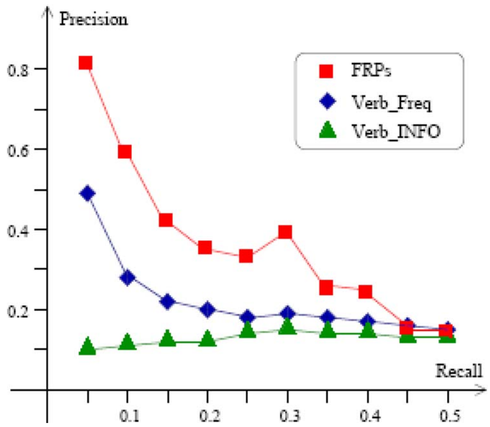


Fig. 6. Precision–recall curves comparing different indexing schemes.

TABLE I
EXPERTS' AGREEMENT

	intra-control group	intra-experimental group	inter-group
# of experts	3	3	6
κ / # of pairs	0.54 / 111	0.76 / 72	0.70 / 630
Interpretation	Moderate agreement	Substantial agreement	Substantial agreement

single-term indices. From Fig. 6, it is clear that, on average, FRPs have 46% better precision than Verb_Freq and 181% better precision than Verb_INFO. This suggests that our extraction results are much more accurate. FRPs therefore can be a good starting point for the stakeholders to understand and discuss the domain.

We observed that, in both groups, experts could not reach 100% consensus during phase 3's meeting. For practical reasons, we set the gold standard to the *intersection* of the control group's experts' results. To examine experts' agreement level, we calculated Cohen's kappa for multiple raters [29]. Table I lists the results. While it is very useful to highlight and discuss experts' different opinions, it is encouraging to note the substantial agreement level between the two groups. This, together with the results presented in Fig. 6, demonstrates the effectiveness of using FRPs to capture the domain's functional requirements assets.

C. Discussion

The results from this initial quasi-experiment should not be seen as final definitive results, but only as an indication of what can be expected from a tool like the FRP extractor. More (controlled) experiments are needed to mitigate threats to validity in our current study, e.g., to randomly allocate subjects, carefully interpret constructs like "functional requirements assets" and assess the effects of confounding variables like group dynamics.

It is interesting to note from Table I that experts in the experimental group had more agreement than those in the control group. One explanation might be that FRPs established a useful common ground. Testing this hypothesis, or further assessing which group is more conductive, requires qualitative inquiries, such as expert (exit) interviews.

Although the automarker product line is relatively small and has modest business goals, it suffices to show the applicability and effectiveness of our approach. Our technique for extracting FRPs is scalable because of the following: 1) The algorithm for identifying LAs is computationally efficient; 2) exploiting available NL processing toolset does not present a considerable overhead; and 3) the extraction process is *summarizing*, which means the output (FRPs) is significantly smaller than the input (the requirements document) [30]. Experimenting with the large-size SRSs of National Aeronautics and Space Administration's family of fault-tolerant system services [31] resulted in a compelling summarizing factor around 200: On average, 101 FRPs were identified for each of the three SRSs whose average size was 20 477 words.

Our setting the gold standard to experts' intersection is a conservative choice. The precision–recall of Fig. 6 would be improved if we chose the experts' *union* to be the gold standard, but the extracted FRPs will inevitably contain irrelevant information or miss relevant information. However, this seeming drawback is really an advantage: Before the FRPs can become a product line's assets, they must be validated by the domain experts. In summary, our experiment showed that FRP extraction increased operation efficiency by an order of magnitude without significantly compromising quality.

V. CASE STUDY

We conducted an exploratory case study [32] with a multidisciplinary organization offering services in many areas of practice, including transportation and systems. Note that the transportation application area has been an important concern among systems engineers and practitioners, e.g., [33]–[35], etc. Our goal was to assess the usefulness and the scope of applicability of our approach. We were interested in exploring how well our approach could perform on a large industrial scale in a real-world setting.

A. Setting

The subject in our study was a set of traffic management systems. Although the group that we collaborated with did not explicitly use the product line idea to manage their products, they were interested in exploring the potential benefits offered by our research. In particular, we collected four related but distinct traffic management SRSs written using the IEEE-830 standard [24]. The average size of the main SRSs is 5884 kB, which is significantly larger than that of the automarker SRSs (293 kB). We intentionally kept the data collection at a *raw* level. In other words, we had little information about the relationships among the four subject systems, e.g., how similar the systems were close to each other, and whether one SRS was used as a baseline for developing the others. It was our intention to address some of these issues via our framework.

We held four meetings with the domain experts during our study. The first meeting was to initiate the collaboration. The second was to know some background information about the subject systems and their SRSs; getting to know some terminologies was also part of the goals of this meeting. The third meeting was to ask the experts to assess the FRP-extraction results. In the last meeting, we presented the FRPs and OVMs produced and collected the experts' feedbacks on both the

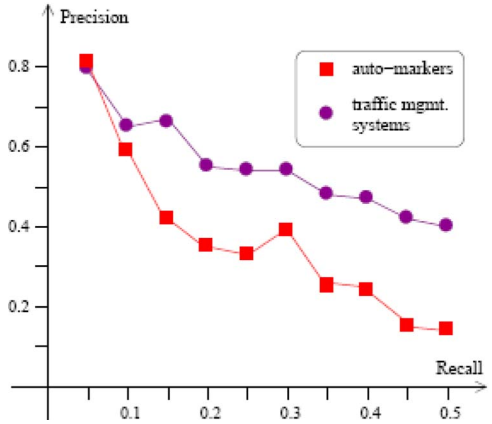


Fig. 7. Precision–recall curves for different FRPs.

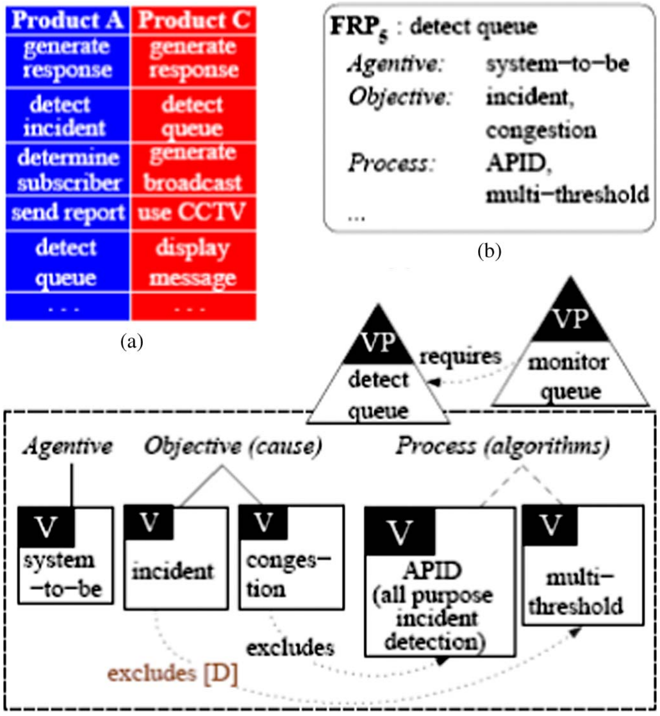
results and our overall approach. Each of the first three meetings lasted about an hour, with one or two experts participating. The last meeting lasted an hour and a half; six experts attended the meeting. We used a mixture of data collection methods, including questionnaires and interviews, in the case study.

B. Results

We followed the experimental group's steps presented in Fig. 5 to evaluate the FRP extraction. Two experts individually validated the FRPs. Then, the experts collaboratively reached a consensus on the domain's functional requirements assets. Following the same procedure described in Section IV-B, we plotted the FRPs' precision–recall curves in Fig. 7. For comparison purpose, Fig. 7 also shows the curve from the automarker study (cf. Fig. 6). Part of the reason that the transportation systems' FRPs outperformed the automarker ones might be the high quality of the industrial SRSs. In practice, we are more likely to obtain results similar to that of the transportation systems' results.

We conducted a focus group session in our fourth meeting with the domain experts. A focus group is a qualitative research method that collects data through group interaction on a topic determined by the researcher [36]. There are typically between 3 and 12 participants in a focus group, and the discussion is guided and facilitated by a moderator-researcher, who follows a predefined questioning structure so that the discussion stays focused.

We devised a questionnaire with eight free-form questions to guide the focus group design. During the focus group's execution, we presented to the participants the top-ranked FRPs [samples from two products are shown in Fig. 8(a)], semantic cases [see Fig. 8(b)], and the OVMs (Fig. 8(c) shows one of the four OVMs presented in the focus group). In addition, we realized that one novel application of our framework was to compare different SRSs via their FRPs. Such an application has much practical value in that profiles play the role of the document's surrogates during reuse candidate identification. We compute two SRSs similarity based on the ratio of the overlapping terms over the distinct total terms. Therefore, we included the SRS similarity matrix (shown in Table II) in the focus group's agenda.



(a)

(b)

(c)

Fig. 8. Sample results presented in the focus group. (a) FRPs. (b) Semantic cases. (c) OVM.

TABLE II
SRS SIMILARITY MATRIX

	SRS_A	SRS_B	SRS_C	SRS_D
SRS_A	1.00	0.78	0.84	0.57
SRS_B	–	1.00	0.75	0.52
SRS_C	–	–	1.00	0.50
SRS_D	–	–	–	1.00

When moderating the focus group, we used PowerPoint presentation and the questionnaire to stimulate the discussion. At predefined breakpoints, we asked the participants to respond to certain questions from the questionnaire and allowed the participants to talk about the topics among themselves. Six domain experts participated in a very lively focus group. We collected three completed questionnaires at the end, all of which were anonymous.

Obviously, the number of sample users of our framework is not representative of the community of systems engineers or requirements analysts, so any quantitative data analysis will lack statistical significance and credibility. However, qualitative data analysis [37] can give an initial reaction to how our approach is considered and perceived by the targeted users. Qualitative data are records of observation or interaction that are complex and contextualized, and they are not easily reduced immediately, or sometimes ever, to numbers. Qualitative research seeks to make sense of the way in which themes and meanings emerged and are patterned in the data records built up from observations, interviews, surveys and questionnaires, and other research media [37]. In our evaluation, we used coding (relating answer sections to proper subject matters under testing) and categorizing (classifying answers to be positive or negative) [37] when analyzing collected data. Table III summarizes our results, where the direct quotes from the respondents are shown in *italic* and cited in double quotation marks (“ ”).

TABLE III
FOCUS GROUP EVALUATION

#	Subject matter	Question (appeared in the questionnaire)	Result	Representative response from participant(s)
1	Scope of the assets	Do you think the model elements (FRPs & semantic cases) capture important domain elements? Are the results surprising? Insightful?	Positive	*"not that surprising"*
2	Product similarity via FRPs	Do you find using the FRP-differences for assessing products' similarity is sensible? Promising?	Positive	*"pretty accurate in IBI's case"*
3	Quality of OVMs	Do you think the commonality, variability, and dependency captured in the OVMs are accurate? Insightful?	Positive	*"known beforehand but a clear way to present"*
4	Triability	According to your experience, do you think this approach (FRPs+OVMs) provides sufficient constructs and guidelines to be tested on a limited basis before adoption?	Somewhat positive	one responded *"yes"*, two suggested additional case studies are needed
5	Observability	Do you see preliminary observable results from the application of the proposed approach to extracting and modeling a domain's requirements assets?	Very Positive	all responded *"yes"*, one further confirmed *"such a tool would be valuable in a company like IBI"*
6	Relative advantage	Compared to relevant techniques you are aware of, do you think that the adoption of the proposed approach can better help you improve the quality of requirements management for a product line?	Inconclusive	*"not sure, but it shows promise"*
7	Complexity	Do you think that the proposed approach is overly complex to be understood and used?	Positive	all responded *"no"*
8	Compatibility	Do you perceive the proposed approach to be compatible and consistent with the existing practices, values, standards, and technologies shared in your organization?	Diverged	one said *"yes"*, another said, *"No. This is radically different."*

Note that questions 4–8 in Table III were designed based on diffusion theory [38], which examines the rate and the motivations of adoption of a technological innovation by a group of potential users. Such an approach may also be fruitful for the evaluation of a novel conceptual framework (such as a design or requirements method) by assessing whether it is appreciated by a community of stakeholders [39]. Therefore, it is crucial for us to apply such a theory to assess whether our framework is appreciated by practitioners, as well as to identify areas for improvement.

C. Discussion

We learned from expert interviews that, because of the similarity often found between the traffic management systems at the requirements level, the systems engineers are tempted to reuse SRSs by "copy-and-paste." Although this appears a natural strategy for producing new (but similar) SRSs, it is inefficient for a number of reasons.

1) Engineers often spend a significant amount of time figuring out the copied requirements' dependences and constraints.
2) The reworking process is not repeatable, nor does it have any general structure which could be easily reapplied.
3) Both correct and incorrect requirements may be reused.

In short, the problem is that requirements are being reused, but in an *ad hoc*, error-prone, and localized manner which does not leverage the full benefit of systematic reuse. An example of reusing incorrect requirements is that only $\mathrm{SRS_D}$ specifies the constraint that incident queue detection should not employ multithreshold algorithms [shown as "excludes[D]" in Fig. 8(c)]. The domain experts confirmed that it is applicable to all the four systems in our study. We conjecture that one of $\{\mathrm{SRS_A}, \mathrm{SRS_B}, \mathrm{SRS_C}\}$ ignored this constraint and the other two reused/inherited such an omission. The speculation is supported by referring to Table II, the FRP-based product-similarity matrix: $\{\mathrm{SRS_A}, \mathrm{SRS_B}, \mathrm{SRS_C}\}$ are close to each other and different from $\mathrm{SRS_D}$.

Our FRP-based extraction and modeling framework has provided the partner company with a taste of what a lightweight approach to product line requirements engineering might look at. We received positive and encouraging results when using the scope, commonality, and variability criteria [16] to evaluate the quality of the OVMs, and the attributes defined in diffusion theory [38] to evaluate whether our overall approach can be spread more widely. The results also suggested areas for improvement, e.g., "relative advantage" and "compatibility."

Although we cannot claim that our work has had a direct impact on the company's existing business-critical systems and requirements engineering process, we can claim to have broadened their views and stimulated process changes. As one expert stated, "*(although) we do not (typically) do R&D. . .there was value looking at this, especially with new eyes.*" Also, the company started planning to run a pilot product line project, not surprisingly on the transportation systems. The goal was to gain experience and make incremental changes.

VI. CONCLUSION

Product lines are rarely created right away, but they emerge when a domain becomes mature enough to sustain their long-term investments. A practical adoption pattern is to build a single system and then build the collection of small variations for the product line [5]. In this paper, we contribute an approach to extracting a product line's requirements assets by scrutinizing the linguistic characterization of a domain's action-oriented concerns and their variabilities. Studies of automarker and traffic management systems show that our automatic support offers an order-of-magnitude saving over manual extraction effort, and our approach receives a positive adoption rate by potential engineers.

Legacy systems and their documentation are valuable source for developing a product line, yet their potential remains largely unexploited [40]. The main thrust of our work is to promote a set of low-threshold techniques as a critical enabler for the systems engineering practitioners to capitalize on the order-of-magnitude improvements offered by product line engineering. Our future work includes providing more automatic support for FRPs' semantic analysis and OVM modeling, exploring novel ways to compute requirements similarity [41], [42], improving the "relative advantage" and "compatibility" aspects of our framework, and incorporating reactive strategies to address evolving requirements [43], [44].

ACKNOWLEDGMENT

The authors would like to thank the partner company for the generous support of our research.

REFERENCES

[1] P. Clements and L. Northrop, *Software Product Lines: Practices and Patterns*. Norwood, MA, USA: Addison-Wesley, 2001.

[2] G. Wang, R. Valerdi, and J. Fortune, "Reuse in systems engineering," *IEEE Syst. J.*, vol. 4, no. 3, pp. 376–384, Sep. 2010.

[3] V. Sugumaran, S. Park, and K. C. Kang, "Software product line engineering," *Commun. ACM*, vol. 49, no. 12, pp. 28–32, Dec. 2006.

[4] D. L. Parnas, "Software product-lines: What to do when enumeration won't work," in *Proc. Int. Workshop VaMoS-Intensive Syst.*, Limerick, Ireland, 2007, pp. 9–14.

[5] C. Krueger, "Eliminating the adoption barrier," *IEEE Softw.*, vol. 19, no. 4, pp. 29–31, Jul./Aug. 2002.

[6] L. Li and J. Liu, "An efficient and flexible web service-based multidisciplinary design optimisation framework for complex engineering systems," *Enterprise Inf. Syst.*, vol. 6, no. 3, pp. 345–371, Aug. 2012.

[7] D. Quartel, M. W. A. Steen, and M. M. Lankhorst, "Application and project portfolio valuation using enterprise architecture and business requirements modeling," *Enterprise Inf. Syst.*, vol. 6, no. 2, pp. 189–213, May 2012.

[8] N. Niu, M. Jin, and J.-R. C. Cheng, "A case study of exploiting enterprise resource planning requirements," *Enterprise Inf. Syst.*, vol. 5, no. 2, pp. 183–206, May 2011.

[9] W. Engelsman, D. Quartel, H. Jonkers, and M. van Sinderen, "Extending enterprise architecture modelling with business goals and requirements," *Enterprise Inf. Syst.*, vol. 5, no. 1, pp. 9–36, Feb. 2011.

[10] M. Fonoage, I. Cardei, and R. Shankar, "Mechanisms for requirements driven component selection and design automation," *IEEE Syst. J.*, vol. 4, no. 3, pp. 396–403, Sep. 2010.

[11] M. Moon, K. Yeom, and H. S. Chae, "An approach to developing domain requirements as a core asset based on commonality and variability analysis in a product line," *IEEE Trans. Softw. Eng.*, vol. 31, no. 7, pp. 551–569, Jul. 2005.

[12] M. Luisa, F. Mariangela, and N. I. Pierluigi, "Market research for requirements analysis using linguistic tools," *Requir. Eng.*, vol. 9, no. 1, pp. 40–56, Feb. 2004.

[13] K. Pohl, G. Böckle, and F. van de Linden, *Software Product Line Engineering: Foundations, Principles, and Techniques*. Berlin, Germany: Springer-Verlag, 2005.

[14] C. J. Fillmore, "The case for case," in *Universals in Linguistic Theory*, E. W. Bach and R. T. Harms, Eds. New York, NY, USA: Holt, Rinehart and Winston, 1968, pp. 1–88.

[15] G. Böckle, P. Clements, J. D. McGregor, D. Muthig, and K. Schmid, "Calculating ROI for software product lines," *IEEE Softw.*, vol. 21, no. 3, pp. 23–31, May/Jun. 2004.

[16] D. M. Weiss and C. T. R. Lai, *Product-Line Engineering: A Family-Based Software Development Process*. Norwood, MA, USA: Addison-Wesley, 1999.

[17] K. C. Kang, S. G. Cohen, J. A. Hess, W. E. Novak, and A. S. Peterson, "Feature-Oriented Domain Analysis (FODA): Feasibility Study," SEI, Carnegie Mellon Univ., Pittsburgh, PA, USA, Tech. Rep. CMU/SEI-90-TR-21, Nov. 1990.

[18] G. Halmans and K. Pohl, "Communicating the variability of a software-product family to customers," *Softw. Syst. Modeling*, vol. 2, no. 1, pp. 15–36, Mar. 2003.

[19] S. Liaskos, A. Lapouchnian, Y. Yu, E. Yu, and J. Mylopoulos, "On goal-based variability acquisition and analysis," in *Proc. IEEE Int. RE Conf.*, Minneapolis, MN, USA, 2006, pp. 79–88.

[20] N. Niu and S. Easterbrook, "So, you think you know others' goals? A repertory grid study," *IEEE Softw.*, vol. 24, no. 2, pp. 53–61, Mar./Apr. 2007.

[21] G. Grieux, J.-C. Léon, F. Guillaume, and N. Chevassus, "Process to integrate design with downstream applications through product shapes adaptation," *IEEE Syst. J.*, vol. 3, no. 2, pp. 199–209, Jun. 2009.

[22] N. Niu and S. Easterbrook, "Concept analysis for product line requirements," in *Proc. Int. Conf. AOSD*, Charlottesville, VA, USA, 2009, pp. 137–148.

[23] N. Niu and S. Easterbrook, "Extracting and modeling product line functional requirements," in *Proc. IEEE Int. RE Conf.*, Barcelona, Spain, 2008, pp. 155–164.

[24] *IEEE Standards Board. IEEE Recommended Practices for Software Requirements Specifications*, IEEE Std 830-1998, 1998.

[25] D. Shepherd, "Natural Language Program Analysis," Ph.D. dissertation, Comput. of Inf. Sci., Univ. of Delaware, Newark, DE, USA, 2007.

[26] Y. S. Maarek, D. M. Berry, and G. E. Kaiser, "An information retrieval approach for automatically constructing software libraries," *IEEE Trans. Softw. Eng.*, vol. 17, no. 8, pp. 800–813, Aug. 1991.

[27] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, 1983.

[28] W. Martin, B. Al, and P. van Sterkenburg, "On the processing of a text corpus: From textual data to lexicographic information," in *Lexicography: Principles and Practice*, R. R. K. Hartmann, Ed. New York, NY, USA: Academic, 1983.

[29] K. Krippendorff, *Content Analysis: An Introduction to Its Methodology*. Newbury Park, CA, USA: Sage, 1980.

[30] A. Mahmoud and N. Niu, "An experimental investigation of reusable requirements retrieval," in *Proc. IEEE Int. Conf. IRI*, Las Vegas, NV, USA, 2010, pp. 330–335.

[31] Mirrored FTSS. [Online]. Available: http://web.mit.edu/16.35/www/project

[32] R. Yin, *Case Study Research: Design and Methods*. Newbury Park, CA, USA: Sage, 2003.

[33] W. H. Ip and D. Wang, "Resilience and friability of transportation networks: Evaluation, analysis and optimization," *IEEE Syst. J.*, vol. 5, no. 2, pp. 189–198, Jun. 2011.

[34] A. Mostafavi, D. M. Abraham, D. DeLaurentis, and J. Sinfield, "Exploring the dimensions of systems of innovation analysis: A system of systems framework," *IEEE Syst. J.*, vol. 5, no. 2, pp. 256–265, Jun. 2011.

[35] A. Stone, "An ontological approach to quantifying the functional flexibility of embedded systems," *IEEE Syst. J.*, vol. 5, no. 1, pp. 111–120, Mar. 2011.

[36] D. L. Morgan, "Focus groups," *Annu. Rev. Sociol.*, vol. 22, no. 1, pp. 129–152, Aug. 1996.

[37] L. Richards, *Handling Qualitative Data: A Practical Guide*. Newbury Park, CA, USA: Sage, 2003.

[38] E. M. Rogers, *Diffusion of Innovations*. New York, NY, USA: Free Press, 1995.

[39] N. Niu and S. Easterbrook, "Analysis of early aspects in requirements goal models: A concept-driven approach," in *Proc. Trans. Aspect-Oriented Softw. Develop.*, 2007, vol. III, pp. 40–72.

[40] I. John, "Integrating legacy documentation assets into a product line," in *Proc. Int. Workshop Product Family Eng.*, Bilbao, Spain, 2001, pp. 113–124.

[41] C. Trummer, C. Ruggenthaler, C. M. Kirchsteiger, C. Steger, R. Weiß, M. Pistauer, and D. Dalton, "Searching extended IP-XACT components for SoC design based on requirements similarity," *IEEE Syst. J.*, vol. 5, no. 1, pp. 70–79, Mar. 2011.

[42] H. Rahnama, A. Sadeghian, and A. M. Madni, "Relational Attribute Integrated Matching Analysis (RAIMA): A framework for the design of self-adaptive egocentric social networks," *IEEE Syst. J.*, vol. 5, no. 1, pp. 80–90, Mar. 2011.

[43] M. A. Corsello, "System-of-systems architectural considerations for complex environments and evolving requirements," *IEEE Syst. J.*, vol. 2, no. 3, pp. 312–320, Sep. 2008.

[44] R. Mietzner, F. Leyman, and T. Unger, "Horizontal and vertical combination of multi-tenancy patterns in service-oriented applications," *Enterprise Inf. Syst.*, vol. 5, no. 1, pp. 59–77, Feb. 2011.

Nan Niu (M'08) received the B.Eng. degree in computer science from the Beijing Institute of Technology, Beijing, China, the M.Sc. degree in computer science from the University of Alberta, Edmonton, AB, Canada, and the Ph.D. degree in computer science from the University of Toronto, Toronto, ON, Canada.

He is an Assistant Professor in Computer Science and Engineering at Mississippi State University, MS, USA. His research interests include software engineering, requirements engineering, software reuse, enterprise systems, and industrial informatics.

Juha Savolainen received the D.Sc. degree from the Aalto University, Espoo, Finland, and the M.Sc. degree from the Helsinki University of Technology, Aalto, Finland.

He is a Software Systems Manager with Danfoss Power Electronics, Graasten, Denmark. His research interests include software engineering, software product lines, requirements engineering, and software architectures.

Zhendong Niu received the Ph.D. degree in computer science from the Beijing Institute of Technology, Beijing China, in 1995.

He was a postdoctoral researcher at the University of Pittsburgh, Pittsburgh, PA, USA, from 1996 to 1998, a research/adjunct faculty member at Carnegie Mellon University, Pittsburgh, PA, USA, from 1999 to 2004, and a joint research professor in the Information School at the University of Pittsburgh from 2006. He is a Professor and the Deputy Dean of the School of Computer Science and Technology at the Beijing Institute of Technology. His research areas include informational retrieval, software architecture, digital libraries, Web-based learning techniques, etc.

Dr. Niu was the recipient of the IBM Faculty Innovation Award in 2005 and the New Century Excellent Talents in University of Ministry of Education of China in 2006.

Mingzhou Jin received the Ph.D. degree from Lehigh University, Bethlehem, PA, USA.

He is an Associate Professor in Industrial and Information Engineering at The University of Tennessee, Knoxville, TN, USA. His research areas include optimization, logistics and transportation engineering, supply chain management, disaster management, systems engineering, and telecommunications.

Jing-Ru C. Cheng received the Ph.D. degree in computer science from Pennsylvania State University, University Park, PA, USA, in 2002.

She has been a Computer Scientist at the U.S. Army Engineer Research and Development Center, Vicksburg, MS, USA, since 2002. Her research interests include parallel algorithm development, software tool development for scientific computing, and multiscale multiphysics code development.