# Advances in Requirements Engineering: Bridging the Gap between Stakeholders' Needs and Formal Designs*

Luqi[1] and Fabrice Kordon[2]

[1] Naval Postgraduate School, Monterey, California, USA luqi@nps.edu
[2] LIP6, Université Pierre et Marie Curie, Paris, France, Fabrice.Kordon@lip6.fr

**Abstract.** The lions's share of the software faults can be traced to requirements and specification errors, so improvements in requirements engineering can have a large impact on the effectiveness of the overall system development process. A weak link in the chain is the transition from the vague and informal needs of system stakeholders to the formal models that support theoretical analysis and software tools.
This paper explains the context for the 2007 Monterey workshop that was dedicated to this problem. It provides the case study that participants were asked to use to illustrate their new methods, and summarizes the discussion and conclusions of the workshop.

## 1 Introduction

**The Monterey Workshop Series** The objective of the entire series of 15 Monterey workshops since 1992 has been to "increase the practical impact of formal methods in computer-aided software development". The workshop seeks to improve software practice via application of engineering theory and to encourage development of engineering theory that is well suited for this purpose.

Previous workshops have reduced the gap between theoretical and practical aspects of software/system engineering and have produced a consensus that the pain of system development could be reduced via computer aid for or automation of software engineering subtasks based on particular theories and various kinds of formal models. A common theme has been to hide theoretical results and complex mathematical ideas inside tools with simple interfaces so that practitioners could use them without the need to fully understand the theory behind them.

However, there has also been general agreement that the pain of development cannot be eliminated completely. No matter what you do, somewhere in the process some people have to think clearly and in detail to reach agreement on what problems should be solved by the software to be developed. Consequently, requirements, response to changes, and human aspects of programming have been identified as potentially fruitful areas for improvement.

* This work was supported in part by ARO grant 45614CI.

**Goal of the 2007 Monterey Workshop** The 2007 workshop is focused on requirements, particularly the process of transforming vague and uncoordinated needs of individual stakeholders into consistent and well defined requirements that are suitable for supporting automated and computer aided methods for engineering subtasks in the development process to follow.

Errors or failures of software-based systems are due to a variety of causes, e.g. misunderstanding of the real world, erroneous conceptualization, or problems in representing concepts via the specification or modeling notations. Precise specification is a key success factor as are communication and the deliberation about whether the specification is right and whether it has been properly implemented. Not all stakeholders are familiar with the formal models and notations employed. Some important requirements might be difficult to quantify and/or express using formal languages, such as the desire that a system should be user-friendly or easily maintainable. Better technologies for requirements analysis should thus be considered.

The majority of requirements are given in natural language, either written or orally expressed. Other requirements might also be visually expressed in terms of figures, diagrams, images or even gestures. Artificial-intelligence approaches might be used to develop prototypes, which can then be re-engineered using more conventional requirements technologies and safety assurance techniques. For example, we might employ large amounts of semantic and statistical data, knowledge bases and theorem provers to infer as much contextual information as possible from the (vague) textual or visual requirements. Then, some extra questions could be raised to system/software stakeholders to point out some fuzzy (or missing) requirements to be refined or some conflicting requirements to be reconciled.

Accurate automatic analysis of natural language expressions has not yet been fully achieved, and interdisciplinary methodologies and tools are needed to successfully go from natural language to accurate formal specifications. Conformance of a system implementation to its requirements requires dynamic and efficient communication and iteration among system stakeholders. It is in supporting this process, and not in supplanting it, that innovative approaches to requirements analysis need to find their proper role.

We want to gain a better understanding of how to deal with natural language as the vehicle from which we derive system/software requirements, how to use intelligent agents as entities to facilitate semi-automatic requirements-documentation analysis, and how to build automatic systems to aid in requirements/specifications elicitation. The overall aim is to exchange ideas for continued research in the intersection of these two areas and to reduce the gap between theory and practice.

A good case study for these issues is to consider how to extract a conceptual model of the goals and requirements of the software needs discussed in a *blog*. As blogs are unstructured natural language, they represent one of the most difficult challenges for natural language processing. All workshop participants have been

requested to use the case study given in Section 4 of this paper to illustrate their work.

## 2 Focus Areas

The three days of the workshop were organized around the following focus areas:

- *Recent Advances in Requirements Engineering.* Feather compares various approached to specification and requirements analysis. Aschauer et al explore success factors for agile requirements analysis. Dinesh et al present an approach for regulatory conformance checking.
- *Human and Linguistic aspects of Requirements Engineering.* Kof addresses identification of goals in stakeholder dialogs. Sawyer examines profiling and tracing of stakeholder needs. Goedecke explores the relation between viewpoints and documents.
- *Computer Aid for Requirements Engineering.* Fe describes how model-driven prototyping can help elicit requirements. Popescu et al explain how automatically created OO models can be used to improve the quality of requirements specifications.

The panels and discussion sections interleaved with the presentations were focused on integrating, balancing, and assessing the various viewpoints presented at the workshop to reach a consensus on where we are, how emerging capabilities for natural language processing and computer aided requirements elicitation methods can contribute, and to identify the best paths forward.

## 3 Workshop Case Study

All workshop participants were asked to use the case study given below to illustrate their work. The participants in the case study discussion are:

- a representative from the Transportation Security Administration (TSA);
- a representative from the Federal Aviation Administration (FAA);
- a representative from Airport screening and security (ASS).

Their discussion on the blog is reproduced hereafter.

| Who | Post |
|---|---|
| **FAA** | We have to ban on airplane passengers taking liquids on board in order to increase security following the recent foiled United Kingdom terrorist plot. We are also working on technologies to screen for chemicals in liquids, backscatter, you know... |
| **ASS** | Technologies that could help might work well in a lab, but when you use it dozens of times daily screening everything from squeeze cheese to Chanel No. 5 you get False Alarms... so it is not quite ready for deployment! |
| **FAA** | Come on! Generating false positives helped us stay alive; maybe that wasn't a lion that your ancestor saw, but it was better to be safe than sorry. Anyway, I want you to be more alert - airport screeners routinely miss guns and knives packed in carry-on luggage. |
| **ASS** | Well... It's not easy to move 2 million passengers through U.S. airports daily. And people can't remain alert to rare events, so they slip by |
| **TSA** | We can deal with it. What if you guys take frequent breaks? And also we are going to artificially impose the image of a weapon onto a normal bag in the screening system as a test. Then screeners learn it can happen and must expect it. Eventual solution will be a combination of machine and human intelligence. |
| **AAS** | Sounds good though we do take breaks and are getting inspected. We do not get annual 'surprise' tests - sometimes we get them everyday; and if a screener misses too may of these consistently, they are sent to training. |
| **TSA** | We have yet to take a significant pro-active step in preventing another attack - everything to this point has been reactive. Somebody hijacks a plane with box cutters? - Ban box cutters. Somebody hides explosives in their shoes? - X-ray shoes, and then ban matches. We are well behind! |
| **FAA** | What do you suggest? Yes, there is an uncertainty. On each dollar that a potential attacker spends on his plot we had to spend $ 1000 to protect. There are no easy solutions. We are trying to federalize checkpoints and to bring in more manpower and technology. |
| **TSA** | We need to think ahead. For instance, nobody needs a metal object to bring down an airliner, not even explosives. Practically everything inside the aircraft is easily flammable, except for the people, so all anyone needs is oxidizer. Do any of the automated screening devices detect oxidizers? Are the human screeners trained to recognize them? |
| **FAA** | Good point. Airlines need to take the lead on aviation security. The corporate response was to market cheap tickets and pass security off on the federal government. Have a trained group of security officers on every flight. Retrain flight attendants as security officers. Forget about passing around the soda and peanuts - that should be secondary. |
| **AAS** | Sir, a lot of airlines are not doing well and are on the Government assistance. Prices go up, baggage get mishandled. There are constant changes in screening rules – liquids/no liquids/3-1-1 rule. Anything radical will not only cost a lot of money but also deter people. I mean an economic threat is also a threat. |
| **TSA** | I think that enforcing consistency in our regulations and especially in their application will be a good thing to do. Another thing is that even if an airline goes bankrupt there are still advantages: bankruptcy makes it easier to rearrange company assets and to renegotiate vendor and supplier contracts. |
| **FAA** | Ok, we had very productive discussion. Now back to work. I want you to come up with some concrete measures based on what we have been talking about. You should finally generate some ROI for that money we have been spending. And do not forget, the examples listed above are not all-inclusive. |

The objective of the case study exercise is to answer the following questions based on the discussion above:

1. What was the topic(s) of the discussion? Have you noticed any contradictions?
2. What are the realistic requirements that FAA suggests for increasing airport security?
3. What long-terms goals should be set by TSA?
4. What concrete changes should be enforced by Airport screening and security?

## 4 Synthesis of Workshop Discussions

The main points that emerged from workshop discussions are the following:

**Getting Unambiguous Specifications** A major focus of the workshop was the transition from informal ideas to formal models, and the associated problems of resolving ambiguities. This transition is an inescapable part of software development because stakeholder needs, which are inherently informal, must be transformed into software, which is inherently formal. Synthesizing an unambiguous model of stakeholder needs is a major part of requirements engineering; another major part is ensuring that this model is accurate.

It was recognized that natural language is an inescapable part of the process, because most of the communication with stakeholders is carried out in natural language. There has been a great deal of past work on requirements engineering that has advocated the use of formal models to represent requirements by creating notations and tools to make such models accessible to a wider audience.

However, this does not avoid the need for resolving ambiguities. Despite all the past advances, it is still the case that most stake holders are unable to write formal models. These models are constructed by specially trained experts, who construct models on behalf of the stakeholders based on their natural language statements. These experts are at risk of subconscious disambiguation; they construct formal models based on their understanding of stakeholders' statements even though their interpretations could be different than what the stakeholders meant. The model builder may not even be aware that there is an interpretation of the natural language other than the first one that comes to mind and was understood [2].

A similar problem applies to approaches that use unambiguous subsets of natural language. These subsets are made unambiguous by rules and restrictions that admit only one interpretation. Subconscious disambiguation in this case can have the reader relying on an understanding and interpretation of the constrained natural language that differs from the one chosen by the rules and used by all of the software tools based on those rules.

The workshop recognized that it is not possible to write unambiguous natural language, and that it is useful to reduce the amount of ambiguity where possible. Some details can be found in [2, 12]. Other suggested approaches included

using fault tolerance strategies to engineer systems that can tolerate ambiguity and to use examples to clarify which interpretation is intended. Examples could be supplied by stakeholders or generated from formal models and checked by stakeholders.

**Ambiguity Management** Sometimes ambiguity may be used to deliberately express disagreements among different stakeholders. In such cases, questions must be raised to get the correct interpretation. The clients may not know the answer, so negotiations or additional information may be needed to get a reliable resolution.

The workshop concluded that natural language processing and aid for resolving ambiguities would be useful, but should be used to support current processes rather than replacing them. Reasons for this include (1) that there is a lot more to requirements engineering than just translating stakeholder statements into formal models and (2) that current accuracies of automated natural language processing are less than 100%.

**Requirement Engineering** Requirements engineering tasks include finding implied but unstated requirements, detecting conflicts between needs of different stakeholders, and resolving such conflicts. Communication gets increasingly difficult as systems scale up. Stakeholder are typically comprised of diverse groups, each of which has its own specialized domain knowledge, jargon, and unique tacit understanding of the problem. Bridging the gaps becomes key to success as complexity increases because each group typically has only a partial understanding of the issues, constraints, possible solutions and cost implications [14, 9].

Accuracy of the requirements engineering process is crucial. Requirements engineering is a critical part of the system development process because requirement errors cost roughly 100 times less to correct during requirement engineering than after system delivery [4]. This imposes extreme constraints on the accuracy of natural language processing and that we might use to derive system requirements. However, natural language processing accuracies are currently in the 90%-92% range, at best [3]. Therefore natural language processing must be augmented with other methods for removing residual errors, and accuracy must be greatly improved if it is to be seriously used for Requirements engineering.

**Towards Computer Aided Requirement Engineering** To be useful, tools must find all possible instances of a problem, and it is acceptable to have some false positives in the warnings and error reports, otherwise engineers will not be able to afford to rely on the results of the tool. Since the delivered system is unlikely to be any better than the requirements, accuracy of the requirements has great importance. Existing manual processes for deriving requirements form informal stakeholder statements therefore incorporate a variety of checking procedures that include reviews, storyboarding, simulation and prototype demonstration, dependency tracing, consistency checking, and many others. Natural

language processing of requirements engineering must be integrated with such checking procedures to achieve needed accuracy.

Accuracy of natural language processing can be improved by specializing the problem to the context of requirements engineering and using the extra information provided by that context. For details, see [3]. Developing tools and methods for augmenting and supporting current requirements engineering processes with tools that incorporate natural language processing appears to be a promising realistic goal, if it is coupled with integration into error checking and correction processes already used in requirements engineering. Total automation of requirements engineering does not appear to be feasible in the foreseeable future, in view of the gap between promises and actual results of AI research of the past several decades. Natural language processing is highly context dependent, both on the subject domain and the questions being asked. In the context of requirements engineering there are an effectively unlimited number of domains.

Given the huge size of requirements documents for real projects, even imperfect heuristic methods that can improve confidence that something important was not overlooked. Some problems that have been explored in detail include identifying goals in stakeholder dialogs [10], support for dealing with requirements changes [1, 8], using shallow natural language processing techniques to aid in synthesizing requirements [13, 11], and requirements validation [6].

To automate the process it is useful to rely on representations and methods for detecting conflicts or unfounded constraints in the requirements (this can be seen as a second focus that emerged from the workshop). Methods based on logic were proposed for checking conformance of requirements to regulations [5]. A notation and method for analyzing conflicts, ambiguities, and imprecision in requirements based on viewpoints of different stakeholders were explored [7]. These directions are promising because they attempt automation of the processes that cannot be effectively done manually when requirements are very complex. The reason for this is that the analyses are non-local in nature and can depend on interactions between wildly separated parts of the requirements. People are effective at analyzing small bits of text in depth, but not at finding widely separated connections in very long documents. Progress in these directions should be possible in the not too distant future.

**Synthesis of Discussions during the Workshop** Traditionally, Monterey Workshops leave a large space to discussion between participants. In 2007, the workshop discussions resulted in the following conclusions:

- End-to-end integration is necessary for all of the component technologies to realize their possible contributions to real software development processes. To achieve this, a necessary step is to clarify the interface between natural language processing and requirements engineering. [3] contains a step toward this goal.
- Domain specific approaches can help natural language processing perform better. Context information such as the goals of the speaker, the speaker's area of expertise, and expected output of the process can narrow the search

space for disambiguation and condition the probabilities governing the most likely interpretations.
- Natural language processing for requirements engineering needs to handle domain specific jargon and acronyms.
- Generating accurate natural language from formal models is easier and more accurate than the reverse process, and can be very helpful for finding errors. However problems with subconscious disambiguation [2] are still present.
- Generating summary descriptions is useful for finding defects, especially errors of omission.
- To have practical impact, automatic methods contributing to the transformation from natural language to formal requirement models have to be faster and more accurate than current manual methods.

## 5 Conclusion

Overarching goals of the rest of the series of Monterey Workshops are to create a shared community-wide articulation of the system/software engineering enablement challenge, reach consensus on the set of intellectual problems to be solved, and create a common vision of how the solutions to these problems will fit together in a comprehensive engineering environment.

The Monterey Workshop has been able to bring the brightest minds in Software Engineering together with the purpose of increasing the practical impact of formal methods for software development so that these potential benefits can be realized in actual practice. In the workshop, attendees and organizers work to clarify what good formal methods are, what are their feasible capabilities, and what are their limits. Overall, the workshop strives to reduce the gap between theory and practice. This has been a slow and difficult process because theoreticians and practitioners do not normally talk to each other, and did not at the beginning of the workshops. This gap has been gradually reduced. In particular, researchers have focused on problems that are relevant to the practitioners, and have helped demonstrate how recent theory can be applied to solve current problems in software development practice.

Here are the workshops:

| N | Year | Theme | Location | Chairs |
|---|---|---|---|---|
| **0** | 1992 | Concurrent and Real-Time Systems | Monterey | Luqi, Gunter |
| **1** | 1993 | Software Slicing, Merging and Integration | Monterey | Berzins |
| **2** | 1994 | Software Evolution | Monterey | Luqi, Brockett |
| **3** | 1995 | Specification-Based Software Architecture | Monterey | Luqi |
| **4** | 1996 | Computer-Aided Prototyping | Monterey | Luqi |
| **5** | 1997 | Requirements Targeting Software and Systems Engineering | Bernried | Broy, Luqi |
| **6** | 1998 | Engineering Automation for Computer Based-Systems | Carmel | Luqi, Broy |

| | | | | |
|---|---|---|---|---|
| **7** | 2000 | Modeling Software System Structures in a Fastly Moving Scenario | Santa Margherita Ligure | Astesiano, Broy, Luqi |
| **8** | 2001 | Engineering Automation for Software Intensive System Integration | Monterey | Luqi, Broy |
| **9** | 2002 | Radical Innovations of Software and Systems Engineering in the Future | Venice | Wirsing |
| **10** | 2003 | Embedded Systems | Chicago | Shatz |
| **11** | 2004 | Compatibility and Integration of Software Engineering Tools | Vienna | Manna, Henzinger |
| **12** | 2005 | Networked Systems | Irvine | Sztipanovits, Kordon |
| **13** | 2006 | Composition of Embedded Systems | Paris | Kordon, Sokolsky |
| **14** | 2007 | Innovations for Requirements Analysis | Monterey | Luqi, Kordon |
| **15** | 2008 | Foundations in Computer Software | Budapest | Dobrowiecki, Sztipanovits |

The 2007 workshop highlighted some differences between generic natural language processing and natural language processing in the context of requirements engineering. Researchers from both communities learned about relevant recent advances from each of the communities and became more aware of the open problems in the gaps between the two fields. It is becoming clear that many software problems originate in the gap between the fuzzy needs of the human stakeholders and the formal models used in software design. This are is gaining increasing attention from the scientific community.

The Monterey workshops have helped focus the attention of the community on many productive directions. For example, since the 1995 workshop identified specification-based architectures as a key means to achieve system flexibility and reuse, there has been a great deal of activity in these areas. A great deal of research has produced architecture description languages and associated analysis methods, there have been commercial advances on "plug and play" hardware and software, adoption of service-based architectures in electronic commerce, and a move toward open architectures in government and defense systems. Currently the practical impact of software architecture is no longer in doubt

We look forward to comparable advances in computer aided requirements analysis in the decade to come.

## Acknowledgments

The Monterey Workshops were initiated under the support of Dr. Hislop at ARO and many others at NSF, ONR, AFOSR, and DARPA. We would like to thank DARPA and NSF for their financial support of the 2007 workshop, NRC for support of two talented postdoctoral fellows Dr. Rodriguez and Dr. Ivanchenko who contributed to the proposal, workshop case study and material for the web page, the program committee chairs Barbara Paech and Craig Martell and committee members for their efforts on reviewing papers and putting together the workshop program, and the local chair Craig Martell for handling endless practical details. All of the workshop participants contributed to the ideas summarized in this paper.

## References


1. T. Aschauer, G. Dauenhauer, P. Derler, W. Pree, and C. Steindl. Could an Agile Requirements Analysis be Automated? In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
2. D. Berry. Ambiguity in Natural Language Requirements Documents: Extended Abstract. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
3. V. Berzins, C. Martell, Luqi, and P. Adams. Innovations in Natural Language Document Processing for Requirements Engineering. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
4. B. Boehm. *Software Engineering Economics*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1981.
5. N. Dinesh, A. Joshi, I. Lee, and O. Sokolsky. Logic-based Regulatory Conformance Checking. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
6. J. Fu, F. Bastani, and I. Yen. Model-Driven Prototyping Based Requirements Elicitation. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
7. M. Goedicke and T. Herrmann. A Case for ViewPoints and Documents. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
8. A. Hoss and D. Carver. Towards Combining Ontologies and Model Weaving for the Evolution of Requirements Models. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
9. D. Kelly. A software chasm: Software engineering and scientific computing. *IEEE Software*, 24(6):120–119, Nov.-Dec. 2007.
10. L. Kof. On the Identification of Goals in Stakeholders Dialogs. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
11. D. Lange. Text Classification and Machine Learning Support for Requirements Analysis Using Blogs. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
12. D. Popescu, S. Rugaber, N. Medvidovic, and D. Berry. Reducing Ambiguities in Requirements Specifications via Automatically Created Object-Oriented Models. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
13. P. Sawyer, R. Gacitua, and A. Stone. Profiling and Tracing Stakeholder Needs. In *Workshop on Innovations for Requirement Analysis: From Stakeholders Needs to*

*Formal Designs*, volume This issue, page TBD, Monterey, California, September 2008. Springer Verlag, LNCS.
14. A. Stone and P. Sawyer. Identifying tacit knowledge-based requirements. *Software, IEE Proceedings*, 153(6):211–218, Dec. 2006.