Goal-Oriented Requirements Enginering: A Roundtrip from Research to Practice

Axel van Lamsweerde

Université catholique de Louvain
B-1348 Louvain-la-Neuve (Belgium)

avl@info.ucl.ac.be

Abstract

The software industry is more than ever facing the challenge of delivering WYGIWYW software (What You Get Is What You Want). A well-structured document specifying adequate, complete, consistent, precise, and measurable requirements is a critical prerequisite for such software. Goals have been recognized to be among the driving forces for requirements elicitation, elaboration, organization, analysis, negotiation, documentation, and evolution.

Growing experience with goal-oriented requirements engineering suggests synergistic links between research in this area and good practice. We discuss one journey along this road from influencing ideas and research results to tool developments to good practice in industrial projects. On the way, we discuss some lessons learnt, obstacles to technonogy transfer, and challenges for better requirements engineering research and practice.

1. Introduction

Goal-oriented requirements engineering (GORE) is concerned with the use of goals for eliciting, elaborating, structuring, specifying, analyzing, negotiating, documenting, and modifying requirements [14]. Such use is based on a multi-view model showing how goals, objects, agents, scenarios, operations, and domain properties are inter-related in the system-as-is and the system-to-be.

Goals are prescriptive statements of intent whose satisfaction requires the cooperation of *agents* (or active components) in the software and its environment. Goals may be organized in AND/OR structures that capture how they are being refined or abstracted. Such structures form the skeleton of goal models; goals there range from high-level, strategic objectives to fine-grained, technical prescriptions that can be assigned as responsibilities of single agents. The latter may be *requirements* on the software-to-be or *expectations* on its environment.

Goals may refer to functional concerns or quality attributes. A functional goal typically captures some maximal set of desired scenarios; it can be established in a clear-cut sense. In a GORE process, functional goals are used to build operational models such as use cases, state machine models, and the like. A quality goal typically captures some preferred behaviors among those captured by functional goals; in general it cannot be established in a clear-cut sense. In a GORE process, quality goals are used to compare alternative options and select preferred ones, and to impose further constraints on goal operationalizations.

Goals, agents, and scenarios are thus intrinsically inter-related; they form a synergistic triangle on which RE activities may be articulated.

GORE processes are in general a mix of bottom-up and top-down sub-processes as goal models are built by asking WHY and HOW questions about source material obtained from interviews, available documents, etc.

2. GORE Research at a Glance

Seminal RE papers already put WHY concerns at the very heart of the RE process [31]; they were thereby echoing earlier system engineering methodologies such as, e.g., participative analysis [24]. Yue was probably the first to argue that the integration of explicit goal representations in requirements models provides criteria for requirements pertinence and completeness [32]. While the AND/OR structuring of goals, their operationalization, and their association with agents were fairly familiar notions in artificial intelligence [26, 23], Feather was probably the first to provide a precise semantic foundation for goal responsibility assignment in multi-agent systems [7].

Two complementary frameworks emerged for goal-based RE: a formal one [5] and a qualitative one [25]. The former was more focussed on goal satisfaction and the systematic building of complete, conflict-free goal-based

Keynote paper, Proc. RE'04: 12th IEEE International Requirements Engineering Conference, Kyoto, September 2004.

requirements models; the latter was more focussed on quality goal satisficing and the qualitative assessment of alternative goal-based requirements models.

A lot of research was then undertaken in the goal-agent-scenario triangle, e.g., to explicitly model agent dependencies in goal satisficing [33], derive goal refinements that are provably complete [6] and realizable by agents [18], derive goal operationalizations [19], identify goals from scenarios [12, 30], handle obstacles to goal satisfaction [28, 1, 13], manage conflicting goals [11], monitor goal violation scenarios at run time [8], negotiate goal-based requirements [3], reuse goal taxonomies and specifications [1, 22], model and reason about security requirements [2, 21, 16], reason about partal goal satisfaction [20], and assess or derive software architectures from goal-based requirements [9, 15].

3. How about GORE Practice ?

Standards such as IEEE Std-830 and books by practitioners such as [4] suggest that goal orientation is not wishful thinking. In fact, GORE projects have been undertaken in various industrial settings.

Domain	System
Telecom	Phone service through TV cable
Air traffic control	Inter-controller communication support (preliminary study)
Air traffic control	Conflict handling between ground and on board collision avoidance systems (preliminary study)
Aerospace	Design of test suites for rocket launch
Steel industry	Integrated production management
Automotive industry	Production scheduling; inter-facility order processing
Publishing	Copyright tracking & management
Press	Newspaper back office system
Food industry	Supermarket discount management
Pharmaceutics	Drug dispatching & tracking
Pharmaceutics	E-learning service for salesmen
Health care	Patient clinical summary
Health care	Hospital emergency service support
Natural language pocessing	Web page translator

Table 1: A sample of KAOS projects at CEDITI

In particular, the KAOS method [14, 17] has been applied in about 20 industrial projects at CEDITI (a UCL university spin-off), in a wide variety of domains, to engineer requirements for fairly different types of systems. The method has also been used to build goal-oriented models for various strategic planning and business process reengineering projects, to reengineer unintelligible requirements documents, and to generate calls for tenders and tender evaluation forms in a large international organization.

Table 1 illustrates the diversity of domains and systems for which KAOS has been used. Table 2 gives an idea of the size of the goal-based requirements models that were constructed in some of these projects (referred to by letters for confidentiality reasons).

Concept type	A	B	C	D	E	F	G
Goal	370	56	141	341	640	171	151
Requirement	160	50	164	256	440	311	108
Agent	80	24	32	8	315	116	21
Entity	240	123	106	102	215	166	127
Association	90	71	48	13	77	126	5
Operation	NA	59	42	NA	86	36	80

Table 2: Number of modelled concepts for systems *A-G*

Those GORE projects had two main deliverables:

- the requirements model composed of the goal, agent, object and operation models, in HTML format,
- a model-driven requirements document, in some prescribed format (e.g., IEEE Std-830 standard).

The size of the requirements document was typically ranging from 100 to 300 pages.

Medium-size projects typically range from 3 to 8 person-months, with 1-2 consultants working over a period of 2-5 months. Table 3 shows a typical distribution of efforts for such projects by RE activity type. Variations on those figures depend on domain expertise, analyst profile, and how well-defined the problem to be solved is. Based on Table 3 we use to estimate the effort required for model building as twice the effort required for stakeholder interviews.

Interviews	*16 %*
Transcripts, summaries, and elicitation from additional sources	27 %
Modeling goals, objects, agents and operations	*33 %*
Model validation with stakeholders, negotiation of alternatives, revision and documentation	9 %
Others	15%

Table 3: Workload distribution by GORE activity type

4. The Need for Effective Tool Support

Several of the research efforts mentioned in Section 2 led to research tool prototypes. For industrial GORE projects, professional tools are required that scale up to the size of project deliverables and can be used by non-experts.

In our case, the successful completion of GORE projects would not have been possible without a fully reworked, professional version of the GRAIL research prototype. The Objectiver toolset provides a GORE model editor, a model browser, a model analyzer, and a requirements document generator [27]. The *model editor* maintains multiple

consistent views of the goal, obstacle, object, agent and operation sub-models through several components: a diagram editor, a textual annotator, an explorer managing hierarchical views with drag-and-drop facilities, and a text editor that allows foreign texts (such as interview transcripts or other source material) to be integrated, edited and hyperlinked to model elements. The *model browser* is typically used during validation meetings; it allows stakeholders to navigate through complex models by clicking on diagrams/annotations, following hyperlinks, zooming in and out, etc. The *model analyzer* allows the analyst to issue predefined or specific queries about the model for completeness/consistency checking; to extract model fragments for sub-model visualization, use case generation, requirements document generation, fragment reuse; etc. The *document generator* produces a requirements document in RTF or PDF format whose section-subsection structure is generated from the goal refinement graph and from templates reflecting company-specific standards. This structure is filled in with a glossary of terms generated from the object model, textual annotations retrieved from the model, and figures selected by the user through drag-and-drop from the goal, object, agent and operation sub-models.

The Objectiver toolset has been recently enriched with formal analysis tools that allow partial GORE model fragments to be animated and model checked [29]. These tools are being experimented on a few mission-critical projects; they assume that goals and operationalizations are formalized, when and where needed, in a real-time linear temporal logic [13].

5. Lessons Learnt From a GORE Research-Practice Roundrip

Our experience has convinced us that RE research, tool development, and practice should be highly intertwined. In most of the projects we were involved so far, the customers agreed that the requirements document produced was incomparably better, thanks to the technology being used, than what they had seen before. Conversely, the techniques, heuristics, and tools resulting from research were significantly refined, simplified and polished over the years thanks to project feedback.

A rich, multi-view model clearly appeared to be the core RE artifact on which elicitation, specification, analysis, negotiation, documentation and evolution is articulated. This proved to be the case both in projects being developed in-house and in projects being outsourced. The model must be comprehensible to stakeholders and decision makers; it has therefore to be made of the right abstractions, supplemented with good concrete examples. According to our experience, the goal-agent-scenario triangle proved to be really effective. In particular, we repeatedly observed that a well-structured goal model provides an ideal communication interface between business managers and software engineers. Decision makers looked at goal models carefully, paying special attention to *alternative* goal refinements, operationalizations and responsibility assignments; they did not care too much about UML object models; annotated goal diagrams were found to be more helpful for focussed brainstorming, validation, negotiation, and decision making than fairly vague use case diagrams.

Goal models turned out to be quite helpful in producing more robust requirements, through obstacle analysis [13], and in identifying and resolving conflicting concerns [11]. While essential in mission-critical projects, goal-based anticipation of what could go wrong was also effective in many other projects to discover missing requirements.

Another appreciated feature of our GORE models was their built-in vertical traceability – from strategic business objectives to technical requirements to precise specifications to architectural design choices. The ability to capture multiple system versions within the same model through multiple paths of the goal AND/OR graph (e.g., the system as-is, to-be, and likely-to-be-next) was felt very helpful in some cases.

The diversity of RE projects in type, size, and focus call for highly flexible technologies. We felt that a "multi-button" method and tool that by default supports graphical and textual specifications, plus formal specifications *only when* and *where needed* for incremental analysis of critical model fragments, is a promising step in that direction. The majority of projects did dot involve formal analysis; even then, however, the informal use of our formal refinement *patterns* [6, 18] proved to be very helpful in guiding the goal refinement/abstraction process, pointing out missing goals, and exploring overlooked alternatives.

In the end, what bothers customers the most is the quality of project deliverables. The model-driven requirements document generated with our tool was perceived as the main success indicator in many projects.

Building "good" GORE models is critical and far from trivial. More support is needed for guiding non-skilled analysts in the elaboration of adequate, consistent and complete models bottom-up (from concrete scenarios and examples) and top-down (from abstract concerns); such support might be provided through additional modeling heuristics, patterns, and dedicated analysis tools allowing the analyst to "play" with model fragments.

Another significant problem raised in several projects was the lack of simple yet *precise* reasoning schemes for

coping with goals that can be satisfied only partially – in *X%* of the cases, say.

6. Challenges

Wide variations in software development practices have been observed [10]. The road from *best* practice to *normal* practice is long and has many obstacles. There are numerous reasons for this, e.g., progress in RE activities are felt to be harder to measure than other software lifecycle activities; the benefits of using RE technologies are felt to be hard to measure as well; investment in such technologies need to be made without guarantee of successful project development (if any); requirements documents are generally perceived as big, complex, outdated, and too far away from the executable products customers are paying for; requirements quality does not tell much about the quality of the executable product.

The experience briefly outlined in this paper may provide some hope that such obstacles can be overcome on a larger scale in the future.

Acknowledgement. Thanks to the many agents involved in the KAOS project at UCL, CEDITI and CETIC as researchers, consultants or tool developers, in particular, D. Ballant, C. Belpaire, S. Brohez, R. Darimont, R. De Landtsheer, E. Delor, D. Genard, D. Janssens, E. Letier, P. Massonet, J.F. Molderez, C. Nève, C. Ponsard, A. Rifaut, J.L. Roussel, P. Stadnik, H. Tran Van, A. Vanbrabant, and L. Willemet. Warm thanks are especially due to Robert Darimont for providing project figures.

References

[1] A.I. Anton & C. Potts, "The Use of Goals to Surface Requirements for Evolving Systems", *Proc. ICSE-98: 20th Intl Conf. on Software Enginering*, Kyoto, April 1998.

[2] A. Anton, J. Earp and A. Reese, "Analyzing Website Privacy Requirements Using a Privacy Goal Taxonomy", *Proc. RE'02 – Intl. Requirements Engineering Conf.*, Essen, Sept. 2002.

[3] B. W. Boehm, P. Bose, E. Horowitz, & M. J. Lee, "Soft ware Requirements Negotiation and Renegotiation Aids: A Theory-W Based Spiral Approach", *Proc. ICSE-17: 17th Intl. Conf. on Software Engineering*, Seattle, 1995.

[4] A. Cockburn, *Writing Effective Use Cases*. Addison-Wesley, 2001.

[5] A. Dardenne, A. van Lamsweerde and S. Fickas, "Goal-Directed Requirements Acquisition", *Science of Computer Programming*, Vol. 20, 1993, 3-50.

[6] R. Darimont & A. van Lamsweerde, "Formal Refinement Patterns for Goal-Driven Requirements Elaboration", *Proc. FSE'4: 4th ACM Symp. on Foundations of Software Engineering*, Oct. 1996.

[7] M. Feather, "Language Support for the Specification and Development of Composite Systems", *ACM Trans. on Pro gramming Languages and Systems* 9(2), Apr. 87, 198-234.

[8] M. Feather, S. Fickas, A. van Lamsweerde & C. Ponsard, "Reconciling System Requirements and Runtime Behaviour", *Proc. IWSSD'98 - 9th Intl. Workshop on Software Specification and Design*, Isobe, IEEE CS Press, April 1998.

[9] D. Gross & E. Yu, "From Non-Functional Requirements to Design through Patterns", *Requirements Engineering Jl.* Vol. 6, 2001, 18-36.

[10] C. Jones, "Variations in Software Development Practices", *IEEE Software*, Dec. 2003.

[11] A. van Lamsweerde, R. Darimont and E. Letier, "Managing Conflicts in Goal-Driven Requirements Engineering", *IEEE Trans. on Software. Engineering*, Nov. 1998.

[12] A. van Lamsweerde & L. Willemet, "Inferring Declarative Requirements Specifications from Operational Scenarios", *IEEE Trans. on Sofware. Engineering*, Dec. 1998.

[13] A. van Lamsweerde & E. Letier, "Handling Obstacles in Goal-Oriented Requirements Engineering", *IEEE Transactions on Software Engineering*, Oct. 2000.

[14] A. van Lamsweerde , "Goal-Oriented Requirements Engineering: A Guided Tour", *Proc. RE'01: 5th Intl. Symp. Req. Eng.*, Aug. 2001.

[15] A. van Lamsweerde, "From System Goals to Software Architecture", in *Formal Methods for Software Architectures*, M. Bernardo & P. Inverardi (eds.), LNCS 2804, Springer-Verlag, 2003.

[16] A. van Lamsweerde, "Elaborating Security Requirements by Construction of Intentional Anti-Models", *Proc. ICSE'04: 26th Intl. Conf. on Software Engineering*, May 2004.

[17] A. van Lamsweerde, *Goal-Oriented Requirements Engineering: From System Objectives to UML Models to Precise Software Specifications*. Wiley, 2005.

[18] E. Letier & A. van Lamsweerde, "Agent-Based Tactics for Goal-Oriented Requirements Elaboration", *Proc. ICSE'02: 24th Intl. Conf. on Software Engineering*, May 2002.

[19] E. Letier & A. van Lamsweerde, "Deriving Operational Software Specifications from System Goals", *Proc. FSE'10: 10th ACM Symp. on the Foundations of Software Engineering*, Charleston, Nov. 2002.

[20] E. Letier & A. van Lamsweerde, "Reasoning about Partial Goal Satisfaction for Requirements and Design Engineering", *Proc. FSE'12: 12th ACM Symp. on the Foundations of Software Eng.*, Nov. 2004.

[21] L. Liu, E. Yu and J. Mylopoulos, "Security and Privacy Requirements Analysis within a Social Setting", *Proc. RE'03: Intl. Requirements Engineering Conf.*, Sept. 2003.

[22] P. Massonet and A. van Lamsweerde, "Analogical Reuse of Requirements Frameworks", *Proc. RE-97: 3rd Int. Symp. on Requirements Engineering*, Annapolis, 1997, 26-37.

[23] J. Mostow, "A Problem Solver for Making Advice Operational", *Proc. AAAI-83*, Morgan Kaufman, 1983, 279-283.

[24] E. Munford, "Participative Systems Design: Structure and Method", *Systems, Objectives, Solutions*, Vol. 1, North-Holland, 1981, 5-19.

[25] Mylopoulos, J., Chung, L., Nixon, B., "Representing and Using Nonfunctional Requirements: A Process-Oriented Approach", *IEEE Trans. on Sofware. Engineering*, Vol. 18 No. 6, June 1992.

[26] N.J. Nilsson, *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, 1971.

[27] http://www.objectiver.com.

[28] C. Potts, "Using Schematic Scenarios to Understand User Needs", *Proc. DIS'95 - ACM Symp. Designing Interactive Systems: Processes, Practices and Techniques*, Univ. Michigan, Aug. 1995.

[29] A. Rifaut et al, "FAUST: Formal Analysis of Goal-Oriented Requirements Using Specification Tools", *Proc. RE'03*, Sept. 2003.

[30] C. Rolland, C. Souveyet & C. Ben Achour, "Guiding Goal Modeling Using Scenarios", *IEEE Trans. Sofware Eng.*, Dec. 1998.

[31] D.T. Ross, K.E. Schoman, "Structured Analysis for Requirements Definition", *IEEE Transactions on Software Eng.*, Vol. 3, No. 1, 1977.

[32] K. Yue, "What Does It Mean to Say that a Specification is Complete?", *Proc. IWSSD-4: 4th Intl. Workshop on Software Specification & Design*, IEEE, Monterey, 1987.

[33] E.S.K. Yu, "Modelling Organizations for Information Systems Requirements Engineering", *Proc. RE'93: 1st Intl Symp. on Requirements Engineering*, IEEE, 1993.