Goal-Based Modeling Of The Formal Specification Process Using Smart Objects*

Vijay K. Vaishnavi,
Martin D. Fraser and Daniel Turk

DEPARTMENT OF COMPUTER INFORMATION SYSTEMS
GEORGIA STATE UNIVERSITY

December, 1997
GSU CIS Working Paper
CIS-97-10
GOAL-BASED MODELING OF THE FORMAL SPECIFICATION PROCESS USING SMART OBJECTS

Vijay K. Vaishnavi¹, Martin D. Fraser², and Daniel Turk³

¹,³ Department of Computer Information Systems
² Department of Mathematics and Computer Science
Georgia State University
Atlanta, Georgia 30303

May 1997

Abstract

The conventional model for developing formal specifications, in languages such as VDM, Z, and CSP, uses natural language specifications as the starting point. This model uses informal means for validating the formal specifications, does not readily support changes in requirements, and does not facilitate process improvement. We propose a new model for the formal specification process which moves the need for validation from the formal specifications phase leftward by constructing formal specifications directly from goals. This places validation in the realm of interactions between the client and the requirements engineer, supports the inevitable change in requirements, and facilitates the improvement of the process itself.

1. The Problem and its Importance

Formal specification methods use mathematically based languages (e.g., VDM, Z, and CSP) and formal techniques for specifying software systems. Formal specifications are important to high-integrity, safety-critical applications in particular and to quality software development in general [12]. There is long-standing general agreement among researchers and practitioners that requirements specification is fundamental to the process-oriented approach to achieving software quality and reliability. As software systems grow in scale and functionality the likelihood of subtle errors becomes much greater, some of which may cause catastrophic financial or human loss [7]. One way of achieving the goal to enable developers to construct systems that operate reliably is by using formal specification methods [5] [6] [16].

Recently, such difficulties as learning notations and poor tool support have been cited as contributing to low usage of formal methods [7]. These difficulties are typical among the reasons that are given for not using formal methods and formal specifications. In the case of formal specification methods, however, we propose that such difficulties are accidental problems encountered in working with formal languages and that there is an underlying essential problem with producing formal specifications, namely, that the conventional model for the formal specification process uses natural language specifications as the starting point for developing formal specifications. We call this model the Natural Language-Based Formal Specification (NLBFS) model.

The NLBFS model uses informal means for validating the formal specifications, does not readily support changes in requirements, and does not facilitate process improvement.

¹ The work of the first and second authors is partially supported by a research grant from the Research Office, Georgia State University.
Sommerville [20] also recommends that natural language specifications are inadequate as stand-alone specifications; we are proposing that they are also inadequate as starting points for formal specifications. Software requirements are developed with requirements engineering techniques and methods that operate on a rich hierarchical structure of goals (and associated information) [1] [8] [17] which can be lost when a hierarchical set of goals is flattened into a natural language requirements specification document. From this document, formal specifications are developed using a variety of strategies which have been modeled in a framework [10]. These strategies and similar other strategies have natural language specification as the axiomatic starting point for formal specification. The current work re-examines this starting point and suggests a new approach.

The strategies in the framework in [10] have been mapped to the maturity levels with related key process areas, and the more mature strategies have been found to have low usage [11]. This indicates a process problem because the Key Process Areas can be used to identify aspects of the process that need to be improved to attain the next higher maturity level. Thus, the focus should be on changing the formal specification process to eliminate impediments to improvement, among which we claim is the NLBFS model.

Three Inherent Problems

The three essential problems inherent in the NLBFS model discussed above are now further elaborated so that the benefits of the proposed new model can be assessed and evaluated.

Quality of Service of the Client Interface

The interposition of the natural language requirements specification between goals and associated information and the formal specification creates a communication pipe that can only pass natural language information. For example, when a formal specification is inspected for faithful representation of client requirements, generally a validation step is performed because the client often, especially in contractual situations, prefers to assess and approve specifications in natural language [23].

The problem with this process is that the communication medium for assessing and approving a formal expression of the specifications is informal expression, the very vehicle that many researchers have characterized as ambiguous and error prone for requirements specification. Furthermore, for analysts and clients to consider multiple natural language expressions of the requirements is repetition of work done in requirements engineering, which creates opportunities for ambiguity and error through reliance on natural language re-expression of the formal specifications.

Response to Changing Goals

The requirements are generated from operational goals which continually change in most modern-day organizations. The overall formal specification model involves many steps between the origins of the needs for the requirements and the formal specifications and thus makes the ability for the specifications to dynamically adapt to the changing requirements questionable. This problem is accentuated when the natural language requirements specification is a necessary part of the process.
**Incremental Process Improvement**

Adapting the existing formal specification process to meet successively higher levels of maturity is not an incremental activity in which small, like improvements can be accrued with correspondingly small increments in effort. For example, the generic strategies [10] for formal specification involve significant changes for moving from one maturity level to the other. Jumps in activity occur as automation is applied further to the left in the life cycle, ever closer to the natural language specification boundary between the operational goals and the formal specification activities. This phenomenon should be expected – increasing difficulty and complexity is encountered as automation of natural language activities need to be integrated to the automated development of formal specifications, as called for in the more mature formal specification strategies [12].

The above considerations lead to the natural language specification as the source of an underlying problem with the conventional formal specification process. This problem is an essential difficulty because its removal can significantly revise long standing concepts about validation, adaptation to change, and process improvement.

2. A New Model

We propose a new model for the formal specification process called the Goal-Based Formal Specification Process (GBFSP) model which moves the need for validation from the formal specifications phase leftward by constructing formal specifications, expressed in such widely used languages as VDM, Z, and CSP, directly from goals. The model consists of:

- a conceptual Goal Mapping Process (GMP) model, whose maturity is stratified by
- a theoretical Mapping Maturity Measurement (MMM) model to stratify the maturity of GMP.

2.1 Goal Mapping Process (GMP) Model

![A Stratified Multi-Level Goal Hierarchy](image)

A multi-level goal hierarchy is represented in Figure 1. Each node in this hierarchy represents a goal which may also include such information as constraints and exceptions. Goals
may be at many levels of abstraction, with broader, more abstract goals being defined near the top, and narrower, more specific goals being defined as one moves toward the bottom of the hierarchy. In fact, high-level strategic goals may lead to mid-level tactical goals, which may lead to low-level operational goals. System requirements derive from operational goals. We can justify and verify requirements and determine whether they are in alignment with goals. Since goals tend to be more stable than requirements over time [1], it can be seen that there may be multiple low-level requirements which would meet the higher-level goals.

Figure 2. Conceptual Goal Mapping Process (GMP) Model

![GMP Diagram]

Figure 2 shows the conceptual GMP model, with “Goals & Goal Structure” as discussed above. The Domain Analyst constructs a Domain Language in which domain concepts, relationships, operations, and constraints can be expressed. Using the Domain Language, the Requirements Engineer expresses application multi-level goal hierarchies (Figure 1). The Formal Specification Specialist provides the necessary information to transform goal hierarchies directly into formal specifications. The combined work of the DA, RE, and FSS meet the need for flexibility in applying formal specifications methods in practice [2]. We address the operationalization of the GMP model in Section 3 after we develop a measurement model to stratify the maturity of the GMP.

2.2 Mapping Maturity Measurement (MMM) Model

The benefits that the GMP model can produce are degrees of reduction of the negative effects of the three inherent problems of using a natural language requirements specifications as the interface between the goal information and the formal specification (discussed in Section 1). A development organization that maps the goal information directly into formal specifications without mediation by a natural language specification can determine if the potential negative effects have been reduced by regular measurement of the maturity of this formal specification process.

We approach developing the Mapping Maturity Measurement (MMM) model for the Goal Mapping Process model through CMM [18] [19]. As Saiedian and Kuzara [19] point out, moving forward on the CMM index entails advancing on all aspects of the software process (which evidently is slow: “approximately 80 percent of companies assessed...are currently at Level 1.”).

Measurement models are often developed from high to low levels of abstraction by starting with a model expressed with abstract, high-level concepts that can be assumed to be understood. To obtain high-level characteristics of the mapping process, we specialize the
Capability Maturity Model (CMM) to produce a model for measuring the maturity of the GMP. For example, Level 1 specializes simply to a mapping process that is operated for each project individually by each analyst as a cognitive activity without widely applicable management status and quality checks.

It is not so simple to focus the CMM software process attributes on the mapping process above Level 1 where more complex activities occur. At each CMM level above Level 1, we adopt the approach in [12] and use the Key Process Areas (KPAs) for each CMM level to identify systematically the mapping process attributes important to acquire to raise the mapping process to that level. Saiedian and Kuzara [19] summarize the Key Process Areas (KPAs) that “identify where an organization must focus to raise software processes to that level.” Table 1, column 2, summarizes these KPAs. Specialization of the CMM for the GMP starts by focusing on those areas (italicized in Table 1, column 2) that overlap the characterizing attributes for that CMM level for the GMP.

### Table 1. Mapping Maturity Measurement (MMM) Model

<table>
<thead>
<tr>
<th>CMM and MMM Level</th>
<th>Key Process Areas</th>
<th>High-Level Attributes of the Goal Mapping Process (GMP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Initial</td>
<td>N/A</td>
<td>- analyst-dependent mapping of the GMP</td>
</tr>
<tr>
<td></td>
<td></td>
<td>- mappings produced depend upon individual analysts</td>
</tr>
<tr>
<td>2 Repeateable</td>
<td>requirements Mgmt</td>
<td>- management control through repeatable milestones or steps in the GMP</td>
</tr>
<tr>
<td></td>
<td>software project planning</td>
<td></td>
</tr>
<tr>
<td></td>
<td>software project tracking and oversight</td>
<td></td>
</tr>
<tr>
<td></td>
<td>software subcontract management</td>
<td></td>
</tr>
<tr>
<td></td>
<td>software quality assurance</td>
<td></td>
</tr>
<tr>
<td></td>
<td>software configuration management</td>
<td></td>
</tr>
<tr>
<td>3 Defined</td>
<td>organization process focus</td>
<td>- define the GMP with the assistance of computerized rules</td>
</tr>
<tr>
<td></td>
<td>organization process definition</td>
<td></td>
</tr>
<tr>
<td></td>
<td>training program</td>
<td>- computer rules:</td>
</tr>
<tr>
<td></td>
<td>integrated-software management</td>
<td>- assure consistent application and operation of the GMP</td>
</tr>
<tr>
<td></td>
<td>software product engineering</td>
<td>- replace routine labor</td>
</tr>
<tr>
<td></td>
<td>intergroup coordination</td>
<td>- remove routine human error</td>
</tr>
<tr>
<td></td>
<td>peer reviews</td>
<td></td>
</tr>
<tr>
<td>4 Managed</td>
<td>quantitative process management</td>
<td>- incorporate intrinsic measurement of agreement with experience, completeness, and consistency to operate the GMP through computerization</td>
</tr>
<tr>
<td></td>
<td>software quality management</td>
<td></td>
</tr>
<tr>
<td>5 Optimizing</td>
<td>defect prevention</td>
<td>- continuous improvement and optimization of the GMP using a computerized learning system:</td>
</tr>
<tr>
<td></td>
<td>technology-change management</td>
<td>- remove waste and error</td>
</tr>
<tr>
<td></td>
<td>process-change management</td>
<td>- manage the incorporation of change</td>
</tr>
</tbody>
</table>

* Saiedian and Kuzara [19]

* Areas italicized overlap the characterizing attributes for the CMM level for the process of mapping goal information into formal specifications

Next the pertinent attributes for the GMP are identified within the focused areas. For example, for Level 2, software project planning, software project tracking and oversight, and software quality assurance focus attention on the CMM attributes of the organization that enable monitoring the status of a project in the goal mapping process of the organization. Such monitoring is prerequisite to planning, scheduling, and costing for subsequent projects. This focus is key to enhancing the GMP at Level 1 in order to move to Level 2 because of the management difficulties of scheduling and costing (planning, tracking, and oversight) and of
reviewing (quality assurance) a GMP that is the sum of the cognitive activities of the analysts involved. Table 1 (row 2, column 3) summarizes these results for Level 2. Similar considerations were carried out to obtain the attributes of the GMP at each level in Table 1. The resulting measurement model, MMM, given by Table 1 (columns 1 and 3), is an ordinal scale measurement model for the GMP.

The attributes of the process of mapping goals to formal specifications above Level 2 (Table 1, column 3) specify an increasing role of computerization for restricting the corresponding software process attributes of CMM. The use of computerization is a hallmark of a capable software process. Computerization of the software process is a way to cast the process and to permit rapid changes in the process while ensuring rapid response from and conformance by the people in the process. Practically, it is very early to expect computerization of all aspects of a software process. There are, however, certain aspects of the general software life cycle, such as computer assisted specification production, that are good candidates for computerization.

### 2.3 Stratifying the Maturity of the GMP with the MMM Model

Operationalization of the MMM model is critical for a development organization to be able to adapt its software process purposely to achieve the benefits from ameliorating the effects of the three problems inherent with NLBFSs (discussed in Section 1). A model is operationalized by using operational definitions that specify the procedure used to measure the high-level concepts involved. Because any measurement must at least perform classification, operational definitions of the high-level attributes in the MMM model are minimally specifications for classifying the attributes measured. For instance, Fraser, Kumar, and Vaishnavi [10] presented a framework of formal specifications strategies which consisted of four broad generic strategies with defining process attributes. We operationalize the MMM model by identifying five broad generic strategies (given in Table 3, column 2) for performing GMP.

<table>
<thead>
<tr>
<th>MMM Level</th>
<th>Generic Strategies That Classify the High-Level Process Attributes at Each MMM Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Initial</td>
<td><strong>Direct-Unassisted Strategy</strong></td>
</tr>
<tr>
<td></td>
<td>• analysts cognitively produce the formal specifications from studying the hierarchy of goal information</td>
</tr>
<tr>
<td>2 Repeatable</td>
<td><strong>Structured-Unassisted Strategy</strong></td>
</tr>
<tr>
<td></td>
<td>• each operational goal is cognitively mapped by the analyst to a formal specification</td>
</tr>
<tr>
<td></td>
<td>• the collection of formal specifications are refined by the analyst into a smoothened set of formal specifications that meet desired quality heuristics</td>
</tr>
<tr>
<td>3 Defined</td>
<td><strong>Structured-Assisted Strategy</strong></td>
</tr>
<tr>
<td></td>
<td>• each operational goal is mapped to a formal specification with computerized rules</td>
</tr>
<tr>
<td></td>
<td>• the collection of formal specifications is refined using computerized quality heuristics</td>
</tr>
<tr>
<td>4 Managed</td>
<td><strong>Measured-Assisted Strategy</strong></td>
</tr>
<tr>
<td></td>
<td>• computerized knowledge-based system supports the mapping of the operational goals to formal specifications and the heuristic refinement of these specifications based on experiential metrics</td>
</tr>
<tr>
<td>5 Optimizing</td>
<td><strong>Advanced Measured-Assisted Strategy</strong></td>
</tr>
<tr>
<td></td>
<td>• an adaptive learning system uses domain dependent analogies to act on data collected from experiential metrics to automatically tune the knowledge-based system used in Level 4</td>
</tr>
</tbody>
</table>
Validation of the Strategies:

We carry out a series of broad deductions to show that the process attributes defining a generic strategy lead to a capability for achieving the process attributes defining the corresponding MMM level in Table 1.

Direct-Unassisted Strategy

The dependence of mappings on individual analysts characterizing MMM Level 1 can be "achieved" by this strategy because formal specifications are produced cognitively by each analyst after studying the goals and their structure.

Structured-Unassisted Strategy

Management control through repeatable milestones or steps needed for MMM Level 2 can be achieved by taking each mapping of an operational goal into formal specifications as well as the final heuristic refinement of the formal specifications as milestones. Rates of completion of milestones can be stored in a baseline database for use in costing, scheduling, and/or reviewing, which provide management control points. The hierarchical goal structure is a common expression of goal information across all problem domains. Also, the heuristic refinement to achieve quality specifications would operate on the formal specifications after they are mapped from the goals and would not be dependent on the goal structure. Thus, these milestones are repeatable and reusable as required for MMM Level 2 because they are defined independent of the project and its problem domain.

Structured-Assisted Strategy

The use of computerized rules stipulated for MMM Level 3 define the mapping and refinement process with this strategy. Consistent application and operation of the process needed for MMM Level 3 can be achieved through the computerization. Computerization can also replace routine labor and remove routine human labor.

Measured-Assisted Strategy

The knowledge-based system in this strategy is based on and codifies measurement of experience, completeness, and consistency. Because this system automates the production of formal specifications, the mapping and refinement process is operated by this measurement through computerization, as required for MMM Level 4.

Advanced Measure-Assisted Strategy

The learning system in this strategy acts on the measurements and can make incremental improvements on and the optimization of the knowledge-based system as indicated by changes in measurement values. Thus, the learning system manages the incorporation of change and as indicated by measurement values, the change can effect the removal of waste and error, which are required for MMM Level 5.

3. Operationalizing GMP Using Smart Objects
The conceptual Goal Mapping Process model can be operationalized in a knowledge-based object-oriented way by using Smart Objects.

A knowledge-based model separates knowledge from how it is used, relies on domain-specific information, and is based on heuristic, rather than algorithmic, processing. The problems of formally specifying dynamically-changing requirements and an incrementally changing specification process (the second and third inherent problems discussed in Section 1) are examples of open systems [13], requiring that an effective formal specification process model incorporate process, task, and goal changes as the process is being followed. This, coupled with the goal-based approach discussed above, suggests a knowledge-based model [4] because of its ease of incremental development, its robustness under change, and support for incorporation of non-monotonic logics (heuristics).

While many problems have been addressed from a knowledge-based perspective, one potential limitation inherent in most knowledge-based approaches is the global non-partitioned nature of the knowledge base. Traditionally the knowledge base is composed of a large number of rules which are selected and processed as needed. The ability to group and partition this knowledge base is not generally very fine-grained. In contrast, object-oriented systems partition knowledge across objects, with each object encapsulating all the details of the information and processing that it represents. This allows for a much finer-grained partitioning than traditional rule-based systems allow. The Smart Object model (discussed below) provides for this capability, and, in addition, supports inheritance and both procedural and rule-based approaches.

Each node in the multi-level goals / requirements hierarchy (Figure 1) can be represented as a Smart Object (SO), each of which is designed to be able to generate formal specifications from the corresponding operational goals.

The Smart Object model [21] describes a model for partitioning rule-based systems in an object-oriented way, and provides a framework for reasoning about the associated goals hierarchy. This reasoning and control framework is known as the monitor (see Figure 3). Fixed control rules support this reasoning. The model describes how information may be shared among Smart Objects. Using the analogy of a computer and its operating system, the fixed control rules

---

**Figure 3: The Smart Object Model**

![Smart Object Model Diagram](image_url)
essentially implement a “runtime environment” in which domain-specific SO models may execute. The domain-specific models define variable control rules for each object which implement the domain’s procedures and exceptions to these procedures, and encapsulate the data which is necessary at each level. These domain-specific SOs may be refined by inheritance, and thus provide for the same multi-level goals-hierarchy described above.

**Figure 4. Operational Goal Mapping Process (GMP) Model**

We propose that Smart Objects provide a natural mechanism for supporting a stratified multi-level formal specification process. The customers’ goals may be defined as Smart Object rules, and the formal specification generation rules and relationships (between multiple goals / requirements) may be defined as Smart Object rules and methods. The top down and bottom up definition and refinement of goals may be implemented through domain level rules and methods, which the monitor enacts in executing SOs. Current and proposed requirements may be automatically evaluated for their alignment with goals.

Figure 4 shows the operational GMP model. The Domain/Smart Object Language (SOL) Specialist has designed and implemented Translation Smart Objects which receive the goals structure input from the Requirements Engineer and translate them into a hierarchy of SOL Application Smart Objects. The SOL/Formal Specification Specialist has created Formal Specification Smart Objects which provide the methods to map a hierarchy of Smart Objects expressing application goal structure (and associated information) into formal specifications. The hierarchy of Application Smart Objects use the inherited methods to create formal specifications at MMM Level 3. At Level 4, the Application Smart Objects contain knowledge in the form of heuristics and experiential metrics to create formal specifications. At Level 5, the hierarchy of Application Smart Objects tune the knowledge and metrics used at Level 4 to optimize the formal specification development process based on domain dependent analogies.

**An Example:**

As a small, artificial example of how the GMP model operates and how it can be represented using Smart Objects at Maturity Level 3 of MMM, consider a niche company that produces square root packages for scientific community and industry. Company executives note that government awards are being made to improve computing in secondary education and
express their desire to move to enter this new market. Requirements Engineers (REs) develop a hierarchy of goals. There is a strategic goal to produce a square root package that would be inexpensive and easy to learn and use. Two operational goals emerge. Limit the input value to a nonnegative real number, which keeps the new product simpler and so less costly than the company's standard product line of square root packages that involve complex numbers. Define the output to be non-negative real and set the accuracy as the package constant 0.001, which permit only one input parameter and non-negative simple fixed-digit decimal output to make the new product easy to learn and use.

The above goals are developed by the RE in the Domain Language developed by the Domain Analyst (see Figure 4). The resulting goal structure is validated by the RE with the client and becomes the input to the Translation Smart Objects (SO-T in Figure 5) that translate the two operational goals into three application Smart Objects (SO-A(pplication), SO-C(ost), and SO-U(se)), with the goals re-expressed in SOL.

Figure 5. A Level 3 Instantiation of the Operational GMP Model (Figure 4) for the Example

```
Input Goals Structure

SO-T

SO-C
\[
\text{input : REAL}
\text{input \geq 0.0}
\]
\[
\text{Mapping}
\text{SQR-C}
\text{ext \text{rd input} : REAL}
\text{pre \text{input} \geq 0.0}
\text{post \text{true}}
\]

SO-A
\[
\text{output : REAL}
\text{output - output < 0.001 \text{; output} \geq 0.0}
\]

Mapping

SO-U
\[
\text{ext \text{rd input} : REAL}
\text{wr output : REAL}
\text{pre \text{true}}
\text{post \text{output} \text{; output} < 0.001}
\]

heuristic rules

Output Formal Specifications

SQR (input : REAL) output : REAL
\text{pre \text{input} \geq 0.0}
\text{post \text{output} \text{; output} < 0.001}
```
The strategy for Level 3 (Table 3) is to use computerized rules to map each operational goal into a formal specification, after which the formal specifications are refined with computerized quality heuristics. Each of SO-C and SO-A maps its goals into a formal (VDM) specification, respectively, SQRT-C and SQRT-U using methods inherited from the Formal Specifications Smart Objects (SO-FS in Figure 5). The Application Smart Object (SO-A) applies heuristic rules to the formal specifications generated by SO-C and SO-U to get the final formal specifications (SQRT). A (VDM) sequence construction is used by the heuristic rules to produce one specification (SQRT) that changes the specification of state variables, which would have had to have been accommodated in secondary school customer programs using the SQRT package, to the specification of an input and an output parameter. Given that the implemented Smart Objects are correct, the formal specifications generated correspond to the operational goals is correct because the formal specifications are constructed from operational goals using computerized rules [15] [3].

4. Merits of the Solution

This solution (GBFSP) addresses the three inherent problems with Natural Language Based Specification model discussed in Section 1 as follows:

(1) GBFSP moves the need for validation from formal specifications phase leftward to the hierarchical goal structure (and associated information such as constraints and exceptions), above MMM Level 2.

For MMM Levels 3 and higher, the formal specifications are derived by automated construction (see also [10]) which is a behavior preserving transformation which, according to Lehman et al. [15] and Blum [3], does not need to be verified.

(2) GBFSP removes the natural language specification as the starting step of formal specification and directly maps goals (and associated information) into formal specifications which allows the goals to be iteratively changed and maintained, and the specifications to be automatically regenerated for MMM levels 3 and higher.

(3) GBFSP takes natural language activities out of the automated development of formal specifications process and, applying automation to the activities in the Key Process Areas needed to advance to the next level above MMM Level 2, achieves incremental process improvement.

Incremental improvement is achieved by moving from Level 3 through Level 5 as well as in the continued operation of the formal specification process at Level 5. The movements from Level 3 through Level 5 add automation to similar activities which is not the case when natural language specifications are used as a starting step and automation must move toward integrating the natural language specifications into the formal specifications process. The movement from Level 3 to Level 4 automates an already automated rule-based process using the knowledge gained through measuring experience with the automated rule-based Level 3 system. The movement from level 4 to Level 5 automates the changes in the knowledge based system of Level 4 that are called for by the continual use of the measurements in Level 4.
5. Related Work

We have presented the Goal Based Formal Specification Process (GBFSP) model which maps goals and associated information directly into formal specifications expressed in formal languages such as VDM, Z, CSP (using the Goal Mapping Process model) at different maturity levels (measured by the Mapping Maturity Measurement model). We also have operationalized the Goal Mapping Process model using Smart Objects [21].

Techniques have been proposed for translating goals into informal or semi-formal requirements specifications (e.g., [1]) and for mapping informal specifications into formal specifications (e.g., [9]). There also has been work reported on a formal methodology, KAOS, which provides a language for requirements elaboration [8] and requirements specification [17]. The Goal Mapping Process model, in contrast, directly maps goal information into specifications expressed in widely used formal languages, such as VDM, Z, and CSP.

Mapping Maturity Measurement model specializes the Capability Maturity Model to get the high-level attributes for the Goal Mapping Process. This is similar to the specialization conducted for the process of mapping informal (natural language) specifications into formal specifications [12]. Strategies for mapping informal specifications into formal specifications have been identified in [10]. The present work draws some inspiration from this work while proposing strategies for operationalizing the Goal Mapping Process model.

The Smart Objects used in operationalizing the Goal Mapping Process model have been proposed in [21] as a vehicle for modeling and designing operations support systems. Smart Objects have also been used recently for modeling workflow management systems [22] and for automated coordination of interorganizational workflows [14].

References


