Application Semiotics Engineering Process

Gang Zhao
STARLab, Department of Computer Science, Vrije Univeristeit Brussel, Belgium
gang.zhao@vub.ac.be

Abstract. As application semantics becomes more complex and dynamic in IT systems, it is necessary to engineer the application semantics in its own lifecycle of development parallel to system engineering. The application semiotics engineering process is under study as a methodology for engineering complex and dynamic business logic in intelligent applications. It stresses the informal specification, the traceability of engineering decisions and need of late-binding to a particular formal language representation and computational paradigm for distributed multidisciplinary collaborative modelling environment. The article describes how the application semiotics is developed in a lifecycle of iterative development.

1. Introduction

As IT system requirements modelling goes beyond data and operational paradigms to the underlying business rationale, there arises the need to explicitly capture the business semantics and deploy it in a system, encapsulated in response to its dynamic changes in business model, process and rationale. The explicit model of business semantics is the corporal knowledge and important parts of software assets.

This paper presents an on-going exploration of an engineering process to model such semantics and how its activity is best organized on the insight from database modeling, knowledge system development, object-oriented and component-based software engineering, domain engineering and ontology research. It first introduces what the application semiotics engineering process (ASEP) is and describes an ontology-based approach to application semiotics. It illustrates the lifecycle and activities of the process and key instruments and approaches of the methodology.

2. Application semiotics engineering process

Semiotics is a science of signs and their interpretation [2, 3, 4, 22]. Here it is used to refer to a system of signs. A semiotic system is a model of human intelligence or knowledge or logic for communication or cognition. As a semiotic, it has three aspects: semantic, syntactic and pragmatic [21, 28]. It presents the conceptualization of ‘subject world’ [13] in well-formed symbolics and specifies how it is interpreted and processed with respect to specific application contexts.

Semiotics engineering is a process of creating a symbolic system. It is similar to the development of computational models of data, process, object, knowledge or ontology. It includes such tasks as scoping, modeling, integration, deployment and maintenance. It takes two fundamental viewpoints of semiotics: the capture and use of application semiotics. The capture seeks for semantic presentation for communication and consensus. It operates on the semantic and syntactic dimensions of semiotics. It is often concerned with model scalability and reusability. The use emphasises the formal representation for processing and reasoning. It is concerned with pragmatics of semiotic models: from which perspective and in what context of application the semiotic model is applied with computational consistency and effectiveness.

Intelligent information systems can be plotted along the capture and use dimensions in terms of specificity and diversity:

![Figure 1. Specificity of capture and diversity of use](image)

The development of intelligent information systems aspires to move from domain specific semiotics and their dedicated use towards domain generic model in versatile applications. The ASEP is intended for modeling complex business rules, application logic and domain knowledge which need either encapsulated for change or separated from conventional software modelling of functional dimensions of IT systems for different development or asset management. It targets systems with rich application semantics, such as knowledge systems, system integration
with divergent and rapidly changing business logic, semantic interface specification of software components or web services, protocols for semantic interoperation of collaborative processes or systems. The ASEP is aimed at the development of corporate or organizational intelligent systems and open services such as knowledge management systems, semantic web services [17].

3. Application semiotics

Application semiotics stresses the need for semiotics originating from and deployed in applications. Its diverse use comes from its design for a family of products [3]. It is not intended as generic semiotics for some intelligent application to adopt or plug into, but flexible semiotics that allows for topological and epistemological variability and partial reusability. The rationale is well stated in the distinction between ‘generic architecture’ and ‘highly flexible architecture’ in Organization Domain Modelling [26].

The application semiotics exhibits important affinity with the structure of natural language. The semiotic potential of the natural language is agreed on and shared for communication. Yet it allows for room of individual creativity. It has stable core but is capable of meeting the need of changes and diversity. The symbolic underspecification is the mechanism that enables the natural language to serve effectively as communication system with limited means for unlimited scenarios and communication acts. Its bounded set of semiotic resources is under-specified semantically, with only generic reference, to achieve unbounded potential of specific reference in a given context. By the same token it provides reusability to enable the versatility of multivariant conceptualization.

3.1. Ontology-based approach

Ontology is an approximate shared semiotic representation of a subject matter. To fulfill the above mentioned requirements, the DOGMA¹ approach [14, 19, 20] to ontology engineering [5] is adopted with intention to create flexible, reusable bounded semiotics for diverse computational purposes for unbounded pragmatic possibilities. It distinguishes two layers of modelling to create lexons and commitments respectively. The underspecification of lexons underpins their reusability across computational tasks, applications and perspectives. The lexon commitment guarantees the specification needed for semantic consistency and well-formedness in a particular application.

3.2. Lexon and Lexon Base

Lexons represent binary relationship between two entities. They are the vocabulary (not terminology) of the application semiotics. Similar to the vocabulary of the natural language, they have ideational purport without reference to specific application or task contexts. They are the potential and means of the semiotic system yet to be contextualized, deployed and fully specified meaningfully. Thus underspecified, they serve as basis for consensus, agreement, reusability and versatility.

A lexon is a quintuple \(<\gamma, t_1, r_1, t_2, r_2>\), where \(\gamma \in \Gamma\) is a context identifier, \(t_1 \in T\) and \(t_2 \in T\) are terms referring to the entities in a semantic relationship, \(r_1 \in R\) and \(r_2 \in R\) are roles in the semantic relationship. \(\Gamma, T\) and \(R\) are strings over an alphabet, \(\Sigma\).

The context identifier, OrderProcessing, indicates an ideational context in which terms and roles become meaningful. The ideational context is externalized by a set of resources, such as documents, graphs, databases. Through this resource, the semantic extension of a lexon is established, communicated, documented and agreed upon among ontology developers. With specified ideational contexts, the lexons are not unspecified: they are not merely syntactic by nature. They are underspecified, representing the type rather than the token in the application domain. They do not include axioms or constraints that guarantee the semantic soundness for computation or reference to particular instances of OrderManager or AccountsManager in an application. They can be reified to represent particular viewpoints: in the above cases, action, data and organizational views of business processes.

<table>
<thead>
<tr>
<th>Action View</th>
<th>Data View</th>
<th>Organizational View</th>
</tr>
</thead>
<tbody>
<tr>
<td>(\gamma, t_1)</td>
<td>(\gamma, \gamma, r_2)</td>
<td>(\gamma, \gamma, \gamma)</td>
</tr>
<tr>
<td>Select</td>
<td>Select OrderSupplier</td>
<td>OrderManager.Select</td>
</tr>
<tr>
<td>Determine</td>
<td>Determine PaymentMethod</td>
<td>AccountsManager.Determine</td>
</tr>
<tr>
<td>Send</td>
<td>Send DeliveryNote</td>
<td>AccountsManager.Send</td>
</tr>
<tr>
<td>Receive</td>
<td>Receive DeliveryNote</td>
<td>Customer.Receive</td>
</tr>
<tr>
<td>Check</td>
<td>Check CustomerStatus</td>
<td>AccountsManager.Check</td>
</tr>
</tbody>
</table>

The use of lexons follows the principle of minimal encoding bias and minimal ontological commitment [7] with no assumptions of formal language representations or how the semiotics is to be structured in data structure. Lexon base is a bag of lexons, unordered and unstructured. It is a potential in terms of which the application semantics is to be architected and feature-constrained. In analogy to natural language system, its semantic coverage is inconsistent, ambiguous, overlapping, contradictory and redundant. It embodies

---

¹ DOGMA stands for Developing Ontology-Guided Mediation for Agents.
multiple subject worlds as well as the multi-dimensional and multi-perspective presentation of the one and same.

3.3. Commitment Statement and Discourse

Lexons become fully specified in the pragmatic context on the commitment layer. The meaning of commitment here is application-specific interpretation of lexons. The processing agent in a given application context commits to a selected set of lexons with constraints and organized in particular networks. It depicts the application-specific tokens or instances of generic types and classes modeled in the lexon base. Here the lexons become fully specified, consistent and unambiguous, specific to particular task or application or service. The ideational context is semantic whereas the application context is pragmatic with specific references, in a given task sequence, for a particular functionality and in a given system context.

A commitment statement is a lexon augmented with application-specific feature constraints. It is of a tripartite structure: theme, transition and rheme. The theme and rheme are filled with the terms in the lexon and transition is one of the two roles. It has a narrower denotation than the subject-predicate-object statement in RDF [23]. The fillers of themes and rhemes are only resource not value as in RDF terms. The names are borrowed from functional schools of linguistics [6, 11] to emphasise the functional, pragmatic and network perspectives of lexons in particular application contexts. A lexon < ABCDE-1-3.2, SaleOffer, CharacterisedBy, Validity, Characterize> can be turned into a commitment statement as follows.

A set of application-specific commitment statements is a projection or view of the lexon base. Each take a particular perspective in the role selected in the transition of the statement. The key word, with, introduces a list of attribute value pairs as constraints of cardinality, reference scheme, etc.

Commitment statements are connected to each other into commitment discourse, using operators such as those of set and logic relationship, sequence and operational procedures.

The example captures the business logic: at the event of the customer sending a purchase request, the customer status must be checked before the payment method is selected. The required product items must also be checked in parallel possibly. Finally notification is sent to customer on the purchase request.

The structural backbone of commitment discourse is based on four main dimensions of network connection:

- Data flow connection between discourses in the form of input and output parameters and variables
- Connections by logical/structural operators among statements and discourses
- Discourse embedding
- Inter-statement feature constraints

4. Life cycle

In recognition of important commonalities with software development methodology, we adopt the RUP life cycle model [12, 15, 16] to phase the seven ASEP activities into inception, elaboration, construction and transition. While the documentation and validation are the activities going through all the phases, there are different degrees of focus on the problem determination, scoping, analysis, development and deployment in each phase. The darker shading indicates our observation of the intensity of work of a given phase in Figure 2.

The inception phase studies the problem space and determines solution strategies. It seeks to scope and cut up the problem space for modeling and activity management. The elaboration phase consolidates the scope of each modelling attempt and produces the structured and detailed analysis of business logic or knowledge in the problem space. The construction phase models ontology and its application from analyses. The transition phase deploys the ontology in an application-specific form or platform. These activities are iterated with phases.

5. Activities and deliverables

The activities and deliverables of the scoping, analysis and development are designed to facilitate collaborative engineering to build consensus among stakeholders and
developers. It stresses the traceability of engineering decisions for effective communication in multi-disciplinary development teams.

5.1. Scoping with stories

Scoping the problem space for modeling application semiotics is a significant stage for effective and focused development, especially in ill-structured multidisciplinary domains. The instrument to manage the scope of application semantics for modeling is stories.

The story is a semantically rich use case that describes a unit of knowledge or business logic, identified in knowledge elicitation. Its purpose is to communicate and document the focus of attention in the semantic domain. It serves similar purpose as motivating scenarios [8] and central role as UML use cases. It consists of

- meta-data about the story authoring,
- purpose to summarize the intention of the story
- settings to specify business context and assumptions
- characters: actors and objects in the story
- episodes to specify declaratively or procedurally parts of business logic or domain knowledge
- annotation for notes

![Figure 3. A story of business processes](image-url)

Figure 3. A story of business processes

5.2. Analysis

The analysis can be likened to drawing a map of the problem space brought to focus by the story. The aim is to create a conceptual model of the part of domain conceptualization under consideration. The strategy is to divide and conquer. The steps are decomposition and elaboration. The model decomposes the conceptualization into hierarchical structures to manage modelling complexity. A familiar example in software engineering is activity decomposition diagram [24]. Below is an example of business process breakdown.

```
BEGIN Accept purchase request
    IF each item is NOT suspended from catalogues, AND IF each item has complete and accurate specification
    THEN customer purchase request is NOT verified
    ELSE customer purchase request is verified
END
```

The analysis is conducted of documentation, manuals, legislation, protocols of elicitation, by business analysts, knowledge analysts and domain experts. The resultant conceptual model is ‘informal scheme’ [1], elaborated in plain and straightforward natural language, in a terminology of particular subject matter.

5.3. Development

The development of application semiotics takes the result of scoping and analysis as input to produce disciplined schemes [1]: a set of lexons and their commitments.

### Developing lexons

Its main tasks are extraction, abstraction and organization. The extraction of lexons is text-based, taking the result from analysis as input. It spots key words and phrases in a given text in a natural language. The exercise is largely linguistic and similar to skip n’ span of the fast reading. The highlighted words in the following two examples are spotted as key words to be considered at the step of abstraction.

```
IF offerers who make a public offering did not give advance notice thereof to stock exchange regulator, attaching the prospectus to be published to the offerers are conducting unauthorized solicitation of investors
```

The highlighted text, as a working document, provides two important services here. One is that it provides a tangible scope of work at a particular time of development. The other is that it becomes a record of decision-making, traceable and visible across time, location and teams.

The abstraction is a process of postulating abstract conceptions of terms, roles and lexons. The extracted
words and phrases are linguistic embodiment of concepts to be modelled. The surrounding text conveys the context for understanding and communication of concepts. Since it is based on the highlighted text, there is a clear borderline imposed on conceptual modelling with an explicit focus of attention.

### Table 2. Examples of lexons

<table>
<thead>
<tr>
<th>Context</th>
<th>Term</th>
<th>Role</th>
<th>Role</th>
<th>Term</th>
</tr>
</thead>
<tbody>
<tr>
<td>D.58.94.1</td>
<td>Offerer</td>
<td>Make</td>
<td>MadeBy</td>
<td>PublicOffering</td>
</tr>
<tr>
<td>D.58.94.1</td>
<td>PublicOffering</td>
<td>SubtypeOf</td>
<td>Offering</td>
<td></td>
</tr>
<tr>
<td>D.58.94.1</td>
<td>AdvanceNotice</td>
<td>Concern</td>
<td>PublicOffering</td>
<td></td>
</tr>
<tr>
<td>D.58.94.1</td>
<td>Regulator</td>
<td>Receive</td>
<td>ReceivedBy</td>
<td>Notice</td>
</tr>
<tr>
<td>D.58.94.1</td>
<td>AdvanceNotice</td>
<td>Include</td>
<td>IncludedBy</td>
<td>Prospectus</td>
</tr>
<tr>
<td>D.58.94.1</td>
<td>Regulator</td>
<td>Publish</td>
<td>PublishedBy</td>
<td>Prospectus</td>
</tr>
</tbody>
</table>

While the abstraction is confined to the text and a bottom up approach to modelling, the organization is a step that goes beyond the current working document, covering multiple ideational contexts. It is devoted to two main purposes. One is to structure the lexons extracted and abstracted bottom up in the previous steps, such as merging or introducing subtyping relationship. The other is to integrate the lexons into existing semiotic systems, such as upper or foundational ontologies.

### Developing commitments

The development of lexons uncovers abstract conceptual types from the story and analysis model. Having established conceptual types, the development of commitments is to come back to the ground, modelling their tokens. While lexons underpins the flexibility and reusability of the application semiotics with under-specification, the commitment is essentially dedicated to the semantically well-formed, fully specified, consistent actualizations of the underlying patterns with respect to a particular task or application. The fully specified semantics in the commitments depends on their pragmatics: tasks and application context. The activity takes a different point of view of the results of scoping and analysis. It seeks to capture specific business or knowledge entities, processes or patterns in terms of lexons, so that specifics can be interpreted, marshalled, organized or interoperated in term of generics.

The development of commitments involves four steps: select, focus, constrain and connect. It first delineates the semantic space by selecting a set of lexons from the lexon base. Each lexon is tokenized into commitment statement with a focus. A lexon, \(<\gamma; t_1, r_1, t_2, r_2>\), can be focused in the form of \([t_1, r_1, t_2]\) or \([t_2, r_2, t_1]\), depending on the choice of the role. The terms and roles are then constrained to refine or confine the semantic references and properties as required in an application context. The commitment statements are finally connected into a network with set, logical and operational operators. Below is an example of the commitments representing data objects in business processes.

### 5.4. Deployment

The development of commitments models in the perspectives of application purposes and tasks. It does not take into account another pragmatic aspect: system context or computational platforms. The deployment stage considers where the application semiotics is to be used and the format needed to deploy the lexons and commitments. The commitments are treated as the specification of application semantics to be handed over to the application system engineer to load in the application systems.

For example, the application semiotics of business processes can be transformed into BPEL4WS, BPML. The commitments of web service description can be translated into OWL or DAML-S.

### 6. Conclusion

ASEP recognises the need a different track of development for engineering application semantics. The complexity of application semantics requires it to be handled in its own iterative development cycle and management, rather as part of conventional software development based on the paradigm of ‘close ontology’ [1] before the main loop or scattered in the main iteration of software development [16]. It is envisaged as parallel track to system engineering track as in DE [27].

Compared with classical knowledge engineering approaches [9, 18, 25], ASEP emphasises domain
ontology modelling, instead of going straight to code knowledge rules from the result of analysis. This extra effort is justifiable with aims of reusability and maintainability of business logic in dynamic multi-dimensional application domains. It is necessary for a development involving distributed multi-discipline teams intending to cover product-lines of systems. It is desirable for applications requiring complex application semiotics such as large scale knowledge systems, where the effective management and visualization of the existing rules is crucial for controlling modelling complexity. In order to put the knowledge/ontology engineering on the basis of disciplined team work, the traceability of modelling decision is stressed from stories through analysis models to lexons and commitments and their deployment. One may dispute its necessity and overhead on the performance of a given developer at particular moments of development, but the over-all benefits for the whole team of collaborative participants and evolution of development in the full life cycle of the project are significant and far reaching. On the other hand, ASEP is not intended as a methodology of ‘high ceremony’ [16] in order to make sure of agility development necessary in distributed multidisciplinary environment of development.

ASEP is intended to bind the development as late as possible to a formal language of knowledge representation and computational paradigms of particular computation semantics such as inferential, denotational or operative semantics. It considers these issues in conjunction with the pragmatics of application tasks and deployments.

Acknowledgement

This study is partially funded by the EU 5th framework program, IST 2001-38248.

References

[8] M. Grüninger, M. S. Fox, Methodology for the design and evaluation of ontologies, IJCAI-95 Workshop on basic ontological issues in knowledge sharing, Montreal, 1995.