Features Need Stories

Sidney C. Bailin

Knowledge Evolution, Inc., 1221 Connecticut Ave. NW, STE 3B, Washington DC 20036, sbailin@kevol.com

Abstract. We present an extension of the notion of feature models. An ontology of stories clarifies the meaning of potentially ambiguous features in situations where there is no unified domain model, and where candidates for reuse have varied provenance.

Keywords. features, domain model, story, ontology, evaluation.

1 Introduction

This paper argues that feature models, while an essential tool for reuse in any domain, are not in themselves sufficient for the capture and communication of knowledge about variability. We argue that features carry a relatively small amount of semantics, if any at all, and that much of the intended semantics remain implicit in naming conventions, or tacit in experience, or undocumented in hearsay. We suggest that the missing semantics, addressing fine-grained questions of context, interface, function, performance, and rationale, can be usefully conveyed through *stories* attached to features.

By *story* we mean something relatively specific, not simply a chunk of text. There are several attributes that make something a story, but one of the key attributes is that of being *situated*. In a story, context is (almost) everything.

Examples of the kinds of stories we are looking for already exist in the user feedback sections of many web-based product catalogues. User feedback contributions, despite their unstructured form, tend to be highly situated and, thus, are often good examples of the notion of *story* we are advocating. Consumer assistance product reviews, on the other hand, are often much less situated; for this reason, they often fall short both in their sense of urgency and in the amount of crucial background information they provide.

We are not simply advocating for stories. That is an argument we have made elsewhere [1]. Nor are we arguing here for a particular kind of feature model: that, too, we have addressed elsewhere [2]. In this paper we are specifically interested in the way in which stories can be used systematically to enrich the semantics of a feature model. At the core of this vision lies interplay between the formal and the informal.

The bulk of the paper consists of a case study that involved the evaluation of alternative technologies for a particular function within a complex system of systems. We argue in Section 1.1 that the evaluation of alternative candidates is the essence of the reuse process. Our goal is not, however, to present a methodology for trade studies, but rather to explore the kind of contextual information that is necessary in order to perform such evaluations, and to draw conclusions from this about the structure and contents of domain models. Our conclusions are applicable to virtually any domain modeling method.

1.1 Features and Engineering Analysis

Many opportunities for reuse occur in situations lacking a well-developed domain model, and in which the resources required for a full-scale domain analysis (not just money but also time) are unavailable. Such opportunities cannot take advantage of feature-based reuse methods [3] and product-line architectures [4]. Yet they are far more systematic than what is often termed *opportunistic reuse*. We will call this *real-world reuse*.

These situations are often opportunities for knowledge reuse: the primary decision is to choose an existing, well-defined technical approach to solve a problem. The technical approach is typically selected

S.H. Edwards and G. Kulczycki (Eds.): ICSR 2009, LNCS 5791, pp. 51–64, 2009.

through an engineering analysis involving goals, priorities, and tradeoffs. Thus, knowledge reuse is closely related to the disciplines of engineering analysis.

Such situations, however, often invite reuse not just of knowledge but of actual software, for the simple reason that the technical approaches are realized in software. Thus, there is a deep connection between software reuse and the process of engineering analysis—evaluating alternative technologies to meet a set of requirements.

Conversely, *any* software reuse decision ought to involve an evaluation of alternatives. At a minimum, the question of make *vs.* buy (develop *vs.* reuse) should be addressed. Beyond that, if there are multiple candidates for reuse, or multiple features that may or may not be selected, then a systematic evaluation of tradeoffs ought to occur.

But real-world reuse, lying somewhere between systematic and opportunistic, illustrates a limitation of feature models. Because it occurs without a unified domain model, the features considered in the tradeoff studies are not necessarily comparable from product to product, let alone from study to study. There is no controlled vocabulary with which to describe and evaluate the alternative solutions.

1.2 Features and Ontologies

In a previous paper we described the use of ontologies to represent KAPTUR-style domain models: problems, alternatives solutions, their features, rationales, and tradeoffs [5]. We noted that ontology mismatches arise when there are many parochial models of a domain rather than one unified model, and we described a process of progressive resolution of mismatches. In the LIBRA method, this process of resolving models-in-conflict is viewed as the very essence of reuse [6].

Aspects of the parochial contexts can be formalized to provide enriched semantics for their respective feature sets. Using ontologies, which are intrinsically open-ended, we can add concepts for disambiguating features that have slightly different meanings in different contexts (see Section 4.1). But there is a point of diminishing returns at which formalization may not be appropriate, at least not until the domain reaches a certain level of maturity. The appropriate medium for capturing this context is, in such cases, narrative.

1.3 What is a Story?

By *story* we mean a communication that speaks of *characters* responding to some *situation*. The characters need not be people; they can be organizations, systems, computers, software programs or even designs. Something about the initial situation is unstable and causes the characters to act. The actions change the situation, and what follows is a succession of situations requiring action, and actions taken in response. This is the *forward movement* of the story. The emphasis is on causation, or, from another point of view, rationale.

Stories are an especially effective mode of communication because they draw the reader into the story world. This only happens, of course, if what is at stake for the characters is also a concern for the reader. When this happens, the immersive nature of the story and its forward movement give the communication an urgency and sense of reality that ordinary communications lack.

2 Case Study

We illustrate our arguments with a real case study that involved selecting software for use in a large, complex system of systems (SoS). The primary tradeoff was a make *vs.* buy decision; but, as we shall see, each of the alternatives involves some form of reuse.

Salient features of the SoS, which will remain anonymous in this paper, include life-critical functions, high cost, and a complex network of stakeholders including multiple engineering and development organizations.

The application we consider here is a relatively small part of the SoS, responsible for transmitting large volumes of structured data between two major SoS components. The data stream contains sequences of fields in many different data types and occurring with different frequencies. Different fields originate from

different sources, and are routed to other component systems belonging to different stakeholders. This is illustrated in Figure 1.

We called this a *compression trade study* because the primary challenge is to transmit the data efficiently by compressing it into a compact form, then decompressing it upon receipt.

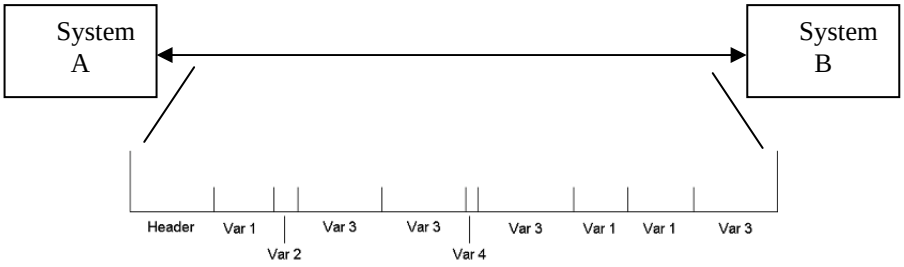

Fig. 1. A data transmission application serves as a case study.

2.1 Evaluation Criteria

Table 1 shows the criteria used to evaluate the alternative compression technologies.

Criterion	**Description**
Encoding / decoding speed	Speed at which messages can be encoded into their compressed format and then decoded by the receiving end
Encoding / decoding software cost	Cost of acquiring and maintaining the software that performs the encoding and decoding
Transmission speed	Direct measure of the size of the compressed messages, which must be averaged over the likely variations in message types and content
Interoperability	Availability of the proposed encoding method on different systems, and the expected level of confidence that the implementations on different systems are compatible
Learning curve / programming complexity	Complexity of an encoding method has a direct impact on the cost of any project-developed software implementing the method, but it may also limit the range of available COTS implementations, and lessen confidence that the implementations are correct
Maturity	Assessed both temporally (how long the method has been in existence) and spatially (how widely it has been used)
Tool availability	Availability of software to perform the selected functions
Tool reliability	The level of confidence that the tool will perform its functions continuously and correctly, where continuity includes the traditional systems engineering notion of availability, while correctness refers to the satisfaction of functional and performance requirements (throughput/response time)
Evolvability	Includes the ability of the selected technology to handle evolution in the data stream contents, and the potential impact of evolution in the compression method itself (as in the evolution of standards)
Risk	General unknowns

Table 1. Evaluation criteria used in the case study.

The evaluation criteria are not mutually independent, but each has a unique emphasis that warrants its being considered explicitly.

2.2 Alternatives Considered

Table 2 identifies the alternative methods that were evaluated:

Solution	Description
Custom packing maps	The format of the data stream messages is defined explicitly in terms of the admissible fields, their sequencing, and their respective length
Abstract Syntax Notation (ASN.1) Packed Encoding Rules (PER)	ASN.1 is an international standard for describing structured data [7]; the PER is part of the standard that provides for a compact binary encoding of such data
Fast Infoset [8]	An international standard for binary encoding of data represented in the Extensible Markup Language (XML)
Lempel-Ziv (LZ) compression [9]	A family of compression technique used by tools such as WinZip and Gzip
Efficient XML Interchange (EXI)	An emerging recommendation of the World-Wide Web Consortium (W3C) for binary encoding of XML data [10]

Table 2. Alternative solutions considered in the case study.

These alternatives represent the gamut from the conventional approach for such systems as previously developed (custom packing maps), through international standards (ASN.1 and Fast Infoset) and widely used file compression techniques (LZ), to a recent W3C recommendation for binary XML transfers (EXI).

The alternatives vary significantly in their levels of maturity, but there is a logic to their consideration. The custom packing map approach is the organization's legacy approach for this type of system. ASN.1 and XML are the two primary standards for expressing structured data outside of programming and formal specification languages. There are several standard encodings for ASN.1, but PER is the primary encoding for efficient binary data transfer. Similarly, Fast Infoset is a current standard for binary encoding of XML, while EXI is an emerging standard for the same purpose, whose definition has itself taken into account many of the alternatives. Finally, LZ compression is widely used in a variety of tools familiar to the engineers responsible for designing and implementing the system.

Further complicating the study is the role of another international standard that would govern the way in which a custom packing map would be specified. To some extent this mitigates the "custom" nature of this alternative (as does the fact that it, alone among the alternatives, draws heavily from the organization's past experience).

3 Reuse in the Case Study

The case study does not sound like a typical reuse scenario. There is no repository of reusable components, and no domain model from which systems can be instantiated through a selection of appropriate features. Nevertheless, the case study illustrates what we are calling *real world* reuse. It is an example of reuse because each of the alternative solutions is a form of reuse. This is clear in the case of the off-the-shelf candidates; perhaps less obvious, but equally true, is reuse in the case of the *custom packing map*, which is the approach the organization had used in all previous systems. At the very least, if a custom packing map were to be adopted, fundamental design concepts from prior systems would be reused; more likely, there would be adaptation of existing source code as well.

Why, then, does the case study appear to be more like a conventional engineering design analysis than a case of reuse? One reason is that the evaluation was performed not at the level of *functionally equivalent alternatives* (for example, alternative implementations of the ASN.1 PER), but rather at the level of *alternative functional approaches*. Formally, however, this is no different from conventional reuse: we have a specification at a certain level of abstraction, and the candidates for reuse are alternative realizations of the specification.

In this case, we lack a unified feature model for characterizing the alternative solutions. But the evaluation criteria serve, in effect, as features. Conversely, in a feature-based reuse process, the features

may be considered evaluation criteria in the selection of a solution. Viewed in this way, every tradeoff analysis creates a kind of domain model, but each model is usually unique to the analysis; they are not easily compared, which makes it difficult to base decisions on prior analyses that were performed in different contexts.

3.1 Features with Values

One objection to viewing evaluation criteria as features is that the criteria are attributes with values, while features are often viewed as things to be selected, i.e., the candidate either has the feature or it does not. This effectively restricts features to attributes with values in a finite enumeration set (such as Boolean-valued attributes). It is an arbitrary limitation that dispenses with information essential to specifying or evaluating reusability.

For example, an efficiency feature could take values in an enumeration set consisting of the time-complexity classes such as Logarithmic, Linear, Polynomial, etc. But it might equally well take the form of more detailed testing results that graph specific response times against different inputs. The studies we consulted follow the latter approach—rightly so, because they tested different classes and distributions of input. The efficiency feature carries implicit semantics that must be made explicit for proper comprehension.

4 Integrating Multiple Feature Models

Having identified the alternative solutions and the criteria for evaluating them, we studied the literature to see what was already known. There was a substantial body of prior work on just this problem, but none of the studies we found used exactly the same set of candidate solutions and none used exactly the same evaluation criteria. The studies varied in their scope, in the level of rigor, and in the detailed definition of features (especially performance features). Not surprisingly, the studies also differed from each other in their conclusions.

As an example, Table 3 lists the criteria from a study by Qing&Adams along with their respective priorities [11].

Criterion	**Priority**
High Compression Ratio	Required
Low Processing Overhead	Required
No Semantic Ambiguity	Required
3rd Party API Support	Desirable

Table 3. Evaluation criteria considered by Qing&Adams.

These criteria resemble ours but are somewhat different. *High compression rate* is effectively synonymous with our *transmission speed* criterion, while low processing overhead corresponds to our *encode/decode speed*. We implicitly required that all the candidates be semantically unambiguous, but we did not specify this explicitly. Finally, their *3rd party API support* could facilitate our *interoperability*, and could improve our *tool availability;* but their criterion is more specific and does not necessarily imply ours.

The alternative solutions considered by Qing&Adams are also similar but not identical to our set, as shown in Table 4. Of the four evaluation criteria in Table 3, the latter two (*no semantic ambiguity* and *3rd part API support*) were quickly disposed of as being satisfied by all candidates, leaving *compression ratio* and *processing overhead* as the focus of the study. Most of the other studies focused on these two criteria, as well. The meaning of the terms, however, varied from study to study.

Solution	**Description**
Gzip and Bzip	Bzip is an alternative compression algorithm using Burrows-Wheeler methods [12]
wbXML	WAP Binary XML, developed by the Open Mobile Alliance [13]

Solution	Description
ASN.1	Apparently the PER encoding although this is not clear
wbXML + Zip	wbXML encoding followed by Gzip or Bzip compression
ASN.1 + Zip	ASN.1 encoding followed by Gzip or Bzip
XML Binary Optimized Packaging (XOP):	W3C recommendation dating from 2005 [14]
Message Transmission Optimization Mechanism (MTOM)	W3C recommendation for binary message transmission especially in the context of the Simple Object Access Protocol (SOAP) [15]
XMill	Open source XML compressor that claims better performance than (for example) Gzip [16]

Table 4. Candidate solutions considered by Qing&Adams.

Now we look at some of the criteria used by the EXI committee. Table 5 shows 7 out of the 21 criteria they applied: those that correspond or are closely related to our criteria [17].

Criterion	Description
Compactness	Amount of compression a particular format achieves when encoding data model items.
Forward compatibility	Supports the evolution of data models and allows corresponding implementation of layered standards.
Implementation cost	How much time does it take for a solitary programmer to implement sufficiently robust processing of the format (the so-called Desperate Perl Hacker measure).
Platform neutrality	Not significantly more optimal for processing on some computing platforms or architectures than on others
Processing efficiency	Speed at which a new format can be generated and/or consumed for processing. It covers serialization, parsing and data binding.
Transport independence	Only assumptions of transport service are "error-free and ordered delivery of messages without any arbitrary restrictions on the message length".
Widespread adoption	Has been implemented on a greater range and number of devices and used in a wider variety of applications.

Table 5. Subset of evaluation criteria used by the W3C EXI committee.

Table 6 illustrates the relationships between the criteria used in our case study and those used by the EXI committee. Some of them—such as *widespread adoption*—are intrinsically “soft” in that they admit of varying interpretations. For these, the value of elaborating the meaning with stories may be obvious. For example, what were the experiences of projects that adopted the technology? To drive our argument home, however, we focus on one particular feature that should, on the face of it, be relatively well-defined: compactness.

Our Criterion	Relation	EXI Criterion
Encoding / decoding speed	Synonymous with	Processing efficiency
Encoding / decoding software cost	Includes off-the-shelf purchase cost as alternative to	Implementation cost
Transmission speed	Effectively synonymous with	Compactness
Interoperability	Enhanced by	Platform neutrality Transport independence
Learning curve / programming complexity	Increases	Implementation cost
Maturity	Indicated by	Widespread adoption

Our Criterion	Relation	EXI Criterion
Tool availability	Indicated by	Widespread adoption
Tool reliability	Weakly indicated by	Widespread adoption
Evolvability	Partially supported by	Forward compatibility

Table 6. Relationships between our evaluation criteria and those of the EXI study.

4.1 Semantics of the Compactness Feature

The term *compactness* is ambiguous because it must cover a wide range of inputs. If we had only one input instance, compactness could be reduced to a positive integer-valued attribute, namely the size of the output. In reality, though, it must represent an aggregate assessment over a set of inputs. Resolving the ambiguity requires deciding on the sample inputs to use, the classes in which to group them, and the kind of aggregation to perform, e.g., averages, distributions, peaks, or some other measure.

For example, Qing&Adams used as input 9 files from the Extensible Access Control Markup Language (XACML) conformance test suite [18], ranging in size from 2KB to 1MB [11]. Mundy *et al* created a custom application for exchanging attribute certificates to compare XML and ASN.1 [19], while Cokus&Winkowsi created test sets to resemble the exchange of binary formatted military messages [20]. The EXI study involved a detailed definition of widely varying use cases [21], while Augeri *et al* created a corpus modeled on the widely used Canterbury corpus for compression evaluations [22, 23].

How do we compare, contrast, and aggregate such various tests? Viewing a feature model as part of an ontology [5], we can create placeholders in which to add the relevant information. For example, the alternative candidates can have a property *test results*, which are then linked to the relevant features of the candidate as well as to the test set employed, as illustrated in Figure2.

This provides a unified view of the past studies. It does not, however, provide much help in comparing the different results. How should one understand the differences in test sets? The EXI tests, for example, correspond to a set of 18 use cases, including Metadata in broadcast systems, Floating point arrays in the energy industry, X3D graphics compression, serialization, and transmission, Web services for small devices, Web services within the enterprise, etc.

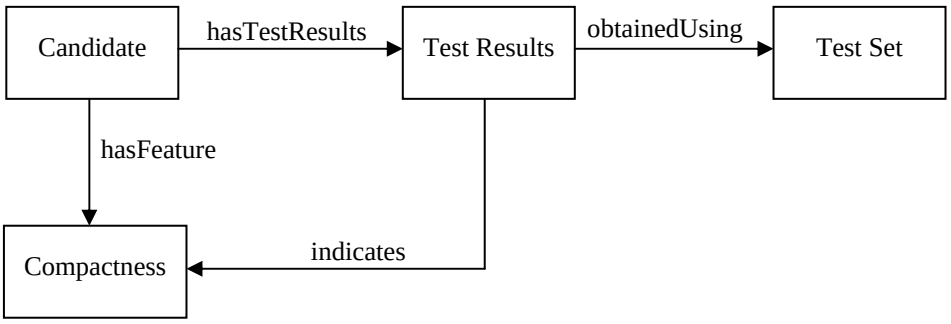

Fig. 2. The open-world character of an ontology allows us to add contextual information to a feature model.

We could add use cases as first-class objects and associate them with the corresponding test sets. Figure 3 illustrates part of the resulting ontology.

We can take this further by elaborating the Scenario class in Figure 3 with properties specifying the functions performed, their relative importance (e.g., required vs. desirable), and their performance attributes. But this does not necessarily help in comparing the results of the different studies. The Canterbury corpus, for example, is not defined in terms of use cases, and its classification of input files is quite unlike the EXI classification. They are really different ontologies. Simply aggregating them provides little insight into the significance of using one test corpus *vs.* another.

A similar issue arises in representing the test results. Some of the studies we consulted present a set of curves—one per candidate—graphing compressed size against original size. But Augeri *et al* average the

results over all test sets and present additional analysis of the test set attributes that were contributing factors [22]. Should these distinctions be modeled in the Test Results class of Figure 2?

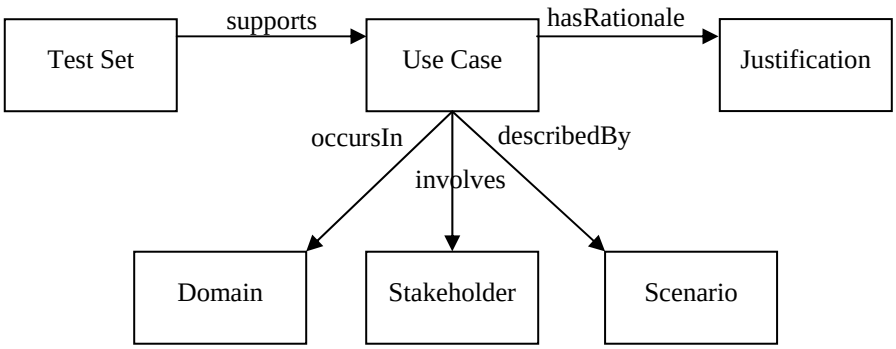

Fig. 3. The feature model can be enriched with an endless amount of contextual information.

A further source of ambiguity is the instrumentation software itself. Different studies may yield different results in part because of different instrumentation software. A discussion in one of the EXI documents illustrates the issues:

> The results presented here come from a stored snapshot that has been judged sufficiently stable for making decisions. Even so, there is still some variation in the results. This is a concern especially with the C-based candidates that may exhibit enormous and inexplicable variance between different test documents. It is therefore likely that there are still some problems with how the test framework's Java code interacts with the C code of the candidates. Accordingly, the results from the C-based candidates were filtered more carefully, by eliminating results for individual documents that were obviously incorrect. In addition, the stableness review of the results paid careful attention to any improvements in these candidates in particular. [24]

We could address this by adding classes of instrumentation, linking them to the test results in a manner similar to Figure 2. But should we? There will always be important contextual information that has not been modeled, that could in principle be modeled, but that would be expensive to model. The alternative is to decorate the feature model with informal elaborations: commentary, explanation, and discussion. The relationships between the formal entities serve to coordinate these informal elaborations into a coherent story that answers questions of the form, Who, Why, What, and How.

5 Elaborating the Feature Model with Stories

Deciding where to draw the line between formalization and informal elaboration must take account of the goals of the model. In our case study, the purpose of the elaborated feature model is to help us choose one of the alternatives listed in Table 2. So let us see how this plays out. Transmission speed being one of the criteria, we consult the Compactness feature, which is effectively equivalent to transmission speed. But Compactness cannot be summarized in a single value. Instead, from the Compactness feature of a given candidate, we can trace back along the *indicates* relationship in Figure 2 to see those test results that tell us something about this candidate's level of success in compressing various inputs. There may be many such test results for this one candidate, including tests by different organizations or individuals, using different test sets, different methodologies, and different instrumentation. Each test result should point to this information.

In the simplest case, a single candidate outperforms all the other candidates, i.e., it consistently produces the smallest output over all test sets in all of the recorded tests. In that case, we can easily say that this candidate scores highest with respect to transmission speed. But of course that is not likely. It is even less likely that we can also identify a clear second-place candidate, third-place candidate, etc., which we would like to do because transmission speed is only one of ten evaluation criteria.

Instead, we have to start asking questions, the answers to which start to provide us with a coherent story. For example:

- Which test sets (if any) resemble our expected data?
- Are the results consistent for those test sets?
- Is the provenance of the test results credible?

If the answers to any of these questions are less than conclusive—for example, if we are uncertain whether any test set closely resembles our data, or the results are not consistent, or the provenance is doubtful—then we have to delve deeper, asking more questions:

- What was the context of a test?
- Who did the test, and what do we know of their agenda?
- Was the methodology rigorous?
- What is the source of inconsistency between this and the tests from other sources?

The answers to such questions may or may not be found in the elaborated feature model shown in Figure 2 and Figure 3, depending on how far one takes the model elaboration. The *questions*, however, are definitely not modeled explicitly. This illustrates a common problem with ontologies: they may be rich in information, but they tend not to be self-explanatory. Many ontologies suffer from an absence of documentation about why the information is there, and how to use it.

This is where stories enter, because the emphasis in a story is on rationale. The persistent question, *Why?* is what distinguishes a story from a mere report. For example, rather than just telling us that a particular test set was modeled on the Canterbury corpus, it would tell us *why* the Canterbury corpus was used. Without knowing this we cannot fairly assess whether the test results are relevant to our problem.

More generally, we would like the elaborated feature model to guide us to the following types of information:

- Who are the stakeholders?
- What are they trying to achieve? Why?
- Why is it non-trivial—what barriers must they overcome?
- What did they do to overcome these barriers?
- Were they successful? Why?
- Did some barriers remain, or new ones appear?

The last three question may be iterated until, eventually, success (or, perhaps, failure) is declared. For example, the Compactness feature for EXI can point us to the URI for the EXI evaluation, which in turn points to earlier documents addressing the above questions. This is the skeletal structure of a story.

But more than that is possible. The questions actually suggest an ontology of stories, consisting (for example) of actors, goals, barriers, actions, and rationales. Discussions about candidates for reuse can be marked up to indicate explicitly the actors, goals, barriers, etc. that have been involved in developing, using, and reusing the candidates. The feature model can use such markup to point us to the information we need. This is illustrated in Figure 4 for the Compactness feature of the EXI alternative.

6 Conclusion

Feature models are a powerful way to achieve reuse in well established domains, such as those admitting of a product line architecture. But much reuse occurs in situations where there is no unified domain model; rather, the candidates for reuse and the available knowledge about them have varied provenance. In such situations, features are less well-defined, and their meanings need to be elaborated through additional context.

Stories, by definition, are situated in context and emphasize rationale. They therefore provide an effective way of enriching a feature model with the necessary context. We have illustrated this through a case study involving the transmission of structured data between two systems.

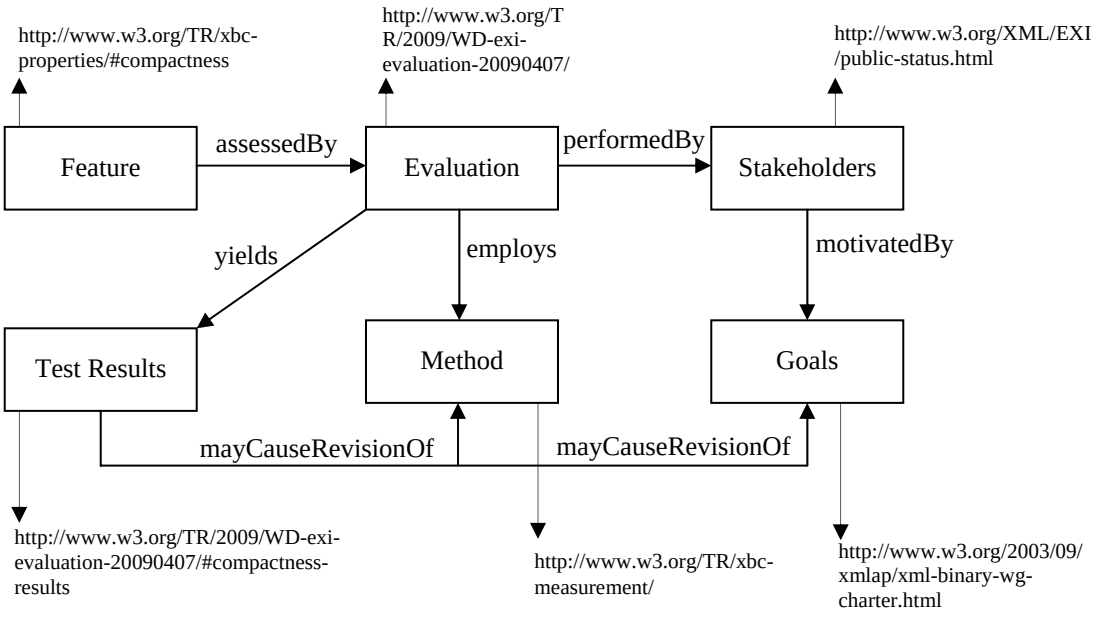

Fig. 4. By incorporating story tags in the elaborated feature model we can guide the potential re-user directly to relevant information.

References

[1] Bailin, S. Diagrams and design stories. *Machine Graphics and Vision*, Vol. 12, No. 1, 2003.

[2] Bailin, S., Moore, M., Bentz, R., and Bewtra, M. KAPTUR: Knowledge acquisition for preservation of tradeoffs and underlying rationales. In *Proceedings of the Fifth Knowledge-Based Software Assistant Conference*. Rome Laboratory, September 1990.

[3] Kang, K., Cohen, S., Hess, J., Novak, W., Peterson, A. *Feature-Oriented Domain Analysis (FODA) Feasibility Study*. Carnegie Mellon University Software Engineering Institute Technical Report CMU/SEI-90-TR-021.

[4] van der Linden, F., Schmid, K., Rommes, E. *Software Product Lines in Action: The Best Industrial Practice in Product Line Engineering*. Springer, 2007.

[5] Bailin, S. Software reuse as ontology negotiation. *Software Reuse: Methods, Techniques and Tools*. Springer LNCS 3107/2004, pages 242-253.

[6] Bailin, S., Simos, M., Levine, L., Creps, R. *Learning and Inquiry-Based Reuse Adoption (LIBRA)*. Wiley-IEEE Press, 2000.

[7] Abstract Syntax Notation: http://www.asn1.org.

[8] Sandoz, P., Triglia, A., Pericas-Geersten, S. Fast Infoset. http://java.sun.com/developer/technicalArticles/xml/fastinfoset/.

[9] Ziv, J. and Lempel; A. *A Universal Algorithm for Sequential Data Compression*, IEEE Transactions on Information Theory, 23(3), pp.337-343, May 1977.

[10] Efficient XML Interchange: http://www.w3.org/XML/EXI/.

[11] Qing, X., and Adams, C. "A comparison of compression techniques for XML-based security policies in mobile computing environments." Ottawa *Workshop on New Challenges for Access Control*, April 27, 2005.

[12] Bzip: http://www.bzip.org/

[13] WAP Binary XML: http://www.w3.org/TR/wbxml/.

[14] XML-binary Optimized Packaging: http://www.w3.org/TR/xop10/.

[15] SOAP Message Transmission Optimization Mechanism: http://www.w3.org/TR/soap12-mtom/

[16] XMill: an Efficient compressor for XML. http://www.liefke.com/hartmut/xmill/xmill.html

[17] Bournez, C (ed.) "Efficient XML Interchange Evaluation". W3C Working Draft, 7 April, 2009. http://www.w3.org/TR/exi-evaluation.

[18] OASIS eXtensible Access Control Markup Language (XACML) TC, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=xacml
[19] Mundy, D.P., Chadwick, D. and Smith, A. “Comparing the Performance of Abstract Syntax Notation One (ASN.1) vs. eXtensible Markup Language (XML)”. *Terena Networking Conference 2008 / CARNet Users' Conference 2008*, May 19-22, 2008, Zagreb, Croatia.
[20] Cokus, M. and Winkiwski, D. “XML Sizing and Compression Study For Military Wireless Data”, *XML 2002*.
[21] XML Binary Characterization Use Cases: http://www.w3.org/TR/2005/NOTE-xbc-use-cases-20050331/.
[22] Augeri, C.J., Bulutoglu, D.A., Mullins, B.E., Baldwin, R.O. “An Analysis of XML Compression Efficiency”. *Proceedings of the 2007 Workshop on Experimental Computer Science*, San Diego, California.
[23] The Canterbury Corpus: http://corpus.canterbury.ac.nz/.
[24] Efficient XML Interchange Measurements Note: http://www.w3.org/TR/2007/WD-exi-measurements-20070725/.