Developing Confidence in Software through Credentials and Low-Ceremony Evidence

Christopher Scaffidi  
Institute for Software Research  
School of Computer Science  
Carnegie Mellon University  
Pittsburgh, PA, 15213 USA  
cscaffid@cs.cmu.edu

Mary Shaw  
Institute for Software Research and  
Computer Science Department  
School of Computer Science  
Carnegie Mellon University  
Pittsburgh, PA, 15213 USA  
mary.shaw@cs.cmu.edu

ABSTRACT
Conventional software specifications and reasoning based on such specifications do not accommodate uncertainty in the specifications, nor do they support the informal, subjective sorts of reasoning that many people use when making decisions about complex systems. We propose a notation for representing specifications in which attributes have different levels of confidence and we discuss ways that uncertain information can contribute usefully to software decisions.

Categories and Subject Descriptors  
D.2.1 [Requirements/Specifications]: Languages, D.2.4 [Design Tools and Techniques], D.2.4 [Software/Program Verification]: Validation

General Terms  
Documentation, Performance, Design, Economics, Reliability, Human Factors, Languages, Verification.

Keywords  
Evolving specifications, credentials, low-ceremony evidence, reasoning under uncertainty

1. INTRODUCTION
Conventional doctrine in software engineering holds that a software component is to be understood from its specification, that the specification is complete and static, and that the specification has been validated through the “high ceremony” techniques of testing, formal analysis, empirical evidence on field use, or perhaps correctness-by-construction for tool-generated components. Unfortunately, this view does not match actual practice very well, especially for large-scale complex systems.

The knowledge on which real developers and users base decisions about software components includes not only the attributes that are typically formally documented in a specification but also attributes such as the reputation of the developers, user opinions, and unsystematic information about performance and reliability. This information changes over time, not always monotonically. Some of the attributes may be validated with high-ceremony evidence, but decisions about software also rely on low-ceremony evidence such as reviews, reputation, advertising claims, qualitative information, or aggregation of group opinion. Individual pieces of low-ceremony evidence often have low credibility, but the accumulation of many pieces of this soft evidence may reasonably provide reasonable levels of confidence. Unfortunately, existing notations and tools do not provide much support for handling this sort of information.

We previously proposed a notation, credentials, for capturing knowledge about a software component together with an indication of the credibility of each value; they support an open-ended set of attributes and can evolve over time [18]. Also, we previously described an approach to combining low-ceremony evidence with high-ceremony evidence and reasoning systematically when evidence has variable degrees of uncertainty [16].

In this paper we propose a refinement of the credentials notation, discuss useful kinds of low-ceremony evidence, and suggest ways that credentials including low-ceremony evidence might help to manage uncertainty.

2. CREDENTIALS
We propose a notation, credentials, that addresses several problems with conventional (static, formal) specifications: Conventional specifications typically focus on a fairly standard set of attributes, but there is little consideration for the cost/benefit tradeoff of actually determining values for those attributes, especially when different clients might be sensitive to different attributes. Credentials are intended to accumulate new or more precise information over time. Further, they are based on property lists, or sets of attribute-value pairs, and are able to incorporate new attributes as they are identified.

Conventional specifications are assumed to be correct; there is no way to indicate the quality of evidence that supports a term of a specification. Credentials explicitly incorporate indications of the credibility of the evidence. They are intended to capture the best information available, understanding that the quality of the information is likely to improve over time but permitting approximate analysis with partial or unreliable information.

Conventional specifications are not designed to incorporate new information about existing attributes as it is discovered; non-monotonic changes are particularly problematic. Credentials can accept new entries and indeed can retain former values of attributes. In addition, credentials include provenance information that can be used to trace dependencies from one credential to another.
2.1 Representation

We propose a representation of credentials as sets of four-tuples <attribute, value, credibility, provenance>, where

- **Attribute** is the name of an attribute. We anticipate that there will be a recognized set of common attribute names and the ability to add new ones, for example for domain-specific attributes. There must be a means to indicate which attributes are required or desired under specified circumstances.

- **Value** is a value for that attribute obtained by analysis, measurement, construction, validation, or perhaps blatant assertion.

- **Credibility** is an indication of the uncertainty associated with the value. It might be drawn from an enumeration such as {default, derived, verified, measured, believed, suspected, claimed}, or it might be quantitative, for example probability distribution functions, ranges, or constraints [16].

- **Provenance** is an indication of the source of the value. For example, if the value is the result of an analysis, the provenance would indicate the analysis and the inputs; if the value comes from a subjective review, the provenance would indicate the review.

This is essentially a property list augmented with metainformation about the quality of the values and about traceability of sources.

Stafford and Wallnau used a similar representation to include a “dossier” containing test harnesses and other software artifacts required for proving property values [20].

2.2 Analysis

Simply representing knowledge about a component or system is not sufficient; the representation should in addition support analysis, or other activities that can take advantage of the information in the representation.

Analysis tools depend on having specific information, and this can be expressed as a condition of applicability: “This analysis can be performed if values for attributes A, B, and C are available; the analysis is more robust if attribute D or E is also available.” Thus the benefit of including an attribute in a specification may depend on the analysis that might be carried out if the value of the attribute is determined.

When -- as in real life -- new information about a system is acquired over time, this information may enable improvements on analyses that have previously been carried out. In most development environments, it is hard to determine which past analyses might be affected. When -- as in real life -- new information contradicts or refutes old information, it is particularly important to know which derived information must be re-examined. The provenance is intended to capture the dependency information required to do this.

When multiple values of an attribute are available, it is often necessary or desirable to reduce them to a single value. For values of different credibility, this requires a calculus of confidence as discussed in [16].

3. LOW-CEREMONY EVIDENCE

A key trait of software engineering is careful selection of components with desired quality attributes. Evidence about components’ attributes comes from different sources. Four have gained widespread acceptance among computer scientists: formal verification, code generation by a trusted automatic generator, systematic testing, and careful empirical studies of the software in operation. We call these “high-ceremony” sources of evidence because, like high-ceremony software development processes [2], these sources of evidence require precise specifications and substantial investment of effort. As explained in detail in [16], high-ceremony sources of evidence may be unavailable or hard/expensive to obtain.

In actual practice, programmers often rely on more subjective, imprecise, or unreliable “low-ceremony” sources of evidence when selecting software components. Potential sources of low-ceremony evidence include:

- reviews of components in professional journals (e.g.:[5]), which recommend certain components for certain contexts, essentially playing the role of a Consumer Reports [4] for professional software developers
- third-party reviews of vendors and products by users (e.g.: amazon.com)
- recommendations by co-workers or friends; popularity
- qualitative reasoning [12]
- advertising claims by vendors
- branding or seller reputation
- certification based on subjective criteria

“best X” reports, often based on linear functions of subjective marks, such as “best schools” or “best doctors”

aggregation of group opinion, obtained statistically or through auction, betting, and other financially-inspired mechanisms for tapping the “wisdom of crowds” [22]

As shown in Table 1, research has identified several low-ceremony sources of evidence that people use outside software.

<table>
<thead>
<tr>
<th>Kind of evidence</th>
<th>Example</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit formal roles</td>
<td>You have to trust your company's lawyer because he is the company lawyer.</td>
<td>[6][9][10][11]</td>
</tr>
<tr>
<td>Prior performance</td>
<td>He has never let you down in the past.</td>
<td>[11][14]</td>
</tr>
<tr>
<td>Models of motivation</td>
<td>He cares about winning your case... so that he gets paid.</td>
<td>[14]</td>
</tr>
<tr>
<td>Informal group membership</td>
<td>He is a member of the Rotary Club.</td>
<td>[6][10][11][23]</td>
</tr>
<tr>
<td>Reputation</td>
<td>“Everybody” knows that he is a good lawyer.</td>
<td>[9][10][11][23]</td>
</tr>
<tr>
<td>References</td>
<td>He said to ask Judy if he is a good lawyer, and she affirms that he is.</td>
<td>[6][9][11]</td>
</tr>
<tr>
<td>Certification</td>
<td>The American Bar Association says that he meets the minimum criteria.</td>
<td>[6][9][20]</td>
</tr>
<tr>
<td>Social context</td>
<td>You can always appeal if this lawyer loses the case.</td>
<td>[8][10][11]</td>
</tr>
</tbody>
</table>
Two of these low-ceremony sources of evidence, reputation and references, already have been used to help professional and end-user programmers select code and components for reuse. For example, the MATLAB Central File Exchange web site records a variety of reputation metrics for posted Matlab functions, including download counters and numeric ratings by end-user programmers [7]. The Mica tool finds Java API functions and code examples by submitting keywords to Google, which uses hyperlinks as a measure of reputation and high-value references [21]. Reputation and references have also been applied to finding spreadsheets [3], web services [24], and general software components [1].

4. MAKING DECISIONS WITH IMPERFECT INFORMATION

Decades of psychological research have shown many ways in which real human decisions do not correspond to the rational analytic derivations that are assumed to be applied to specifications [8]. For example, people make systematic errors in frequency estimates, they misunderstand percentages, and their understanding of risks is driven more strongly by anecdotes than by statistics; memory effects cause recently viewed information to dominate collected facts; and simple linear models often outperform experts.

Given the known limitations in human decision-making, we argue that less-than-perfect information may in many cases be good enough for the quality of reasoning – that is, that low-ceremony data may often not be the weakest link in the chain of reasoning. In other cases, the sensitivity of the result to the values of certain attributes may be low enough that approximate values are good enough [17]. Given that approximate values may be much easier to obtain than precise values, the cost-effectiveness of analysis may be improved with the less expensive data.

Given that real people use low-ceremony evidence routinely for significant decisions, we believe that it is worth trying to incorporate this sort of information systematically in the information used for software decisions.

5. ACKNOWLEDGMENTS

This work was funded in part by the National Science Foundation (ITR-0325273) via the EUSES Consortium and under Grants CCF-0438929 and CNS-0613823.

6. REFERENCES


