What's the PREMES behind your Pattern?

JAN DE MUIJNCK-HUGHES, University of St Andrews
ISHBEL M. M. DUNCAN, University of St Andrews

Design patterns are supposed to be the well documented, tried and tested solutions to recurrent problems. Current evaluation techniques do not provide a demonstrable and holistic means to evaluate pattern quality. This paper introduces *Pattern Report Cards* an evaluation process for software design patterns that is demonstrable, measurable, and reproducible. A set of quality indicators for determining pattern quality has been identified, and a set of qualitative and quantitative evaluation techniques assembled to determine the quality of adherence to these indicators. Further, management and execution of the evaluation process is controlled by the PREMES framework. This framework describes a management cycle that facilitates the construction of bespoke evaluation systems for design patterns. Process tailoring is achieved by providing guidance over the selection and construction of the techniques used to assess pattern quality. Use of these techniques will help bolster existing evaluation processes, and lead to the improvement of design pattern evaluation techniques.

Categories and Subject Descriptors: D.2.4 [**Software**]: Software/Program Verification—*Validation*; D.2.11 [**Software**]: Design—*Patterns*

General Terms: Documentation
Additional Key Words and Phrases: Design Pattern Evaluation, Pattern Evaluation Management System, Design Pattern Quality

ACM Reference Format:

Jan de Muijnck-Hughes and Ishbel M.M. Duncan 2015. What's the PREMES behind your Pattern? HILLSIDE Proc. of Conf. on Pattern Lang. of Prog. 22 (October 2015), 15 pages.

1. INTRODUCTION

First described in Alexander et al. [1977], *Design Patterns* are an engineering technique developed for architecture in which well documented solutions are presented for particular recurrent problems that occur consistently within a well defined context. Large complex problems that cannot be addressed using a single pattern are treated using *Pattern Languages* that document how related sets of patterns are combined to solve these larger problems. Within the domain of software engineering design patterns (and languages) are used to present good solutions to recurrent software engineering problems. Software design patterns are beneficial when promoting the use of *Security Design Patterns* for ensuring the deployment of security mechanisms within software systems [Delessy and Fernández 2008]. However, when creating software design patterns it is unclear over how assurances are to be made towards the quality of the patterns and pattern languages.

A commonly used technique for pattern evaluation is that of *Shepherding*, in which pattern authors are paired with an experienced pattern writer who provides guidance during the creation of the pattern document [Harrison 1999]. However, the shepherding process is a generalised technique applicable to all patterns. This lack of specialisation prohibits a detailed analysis towards a pattern to be given. Although there are existing alternate

This research was supported by an EPSRC Doctoral Training Grant.
The authors can be contacted as follows: Jan de Muijnck-Hughes `jfdm@st-andrews.ac.uk`; and Ishbel M.M. Duncan `ishbel.duncan@st-andrews.ac.uk`.

evaluation techniques for software design patterns, many of them analyse different areas of a pattern. Not all provide a complete treatment of pattern quality according to what a pattern is supposed to be, and not all provide a reproducible means of assessment.

Examining the landscape for security design patterns alone, it becomes apparent that there is a great variance in the quality of patterns being published. For example, Heyman et al. [2007] detail that not all patterns are patterns; and Bunke et al. [2011] that, among other things, not all patterns adhere to common pattern templates. There is a need for a comprehensive solution to software design pattern evaluation that allows for the quality of such patterns to be ascertained.

1.1 Contribution

This paper presents an evaluation framework for software design patterns[1] that allows for pattern quality to be made a reproducible and measurable assessment within the pattern engineering process. Building on existing work the following contributions are made:

(1) We present a set of quality indicators for pattern quality.

(2) We detail *Pattern Report Cards*, a system for the bespoke evaluation of software design patterns using the presented indicators, and metrics gathered from qualitative and quantitative evaluation techniques.

(3) We present a management system based upon the Plan-Do-Check-Act cycle to manage the creation and execution of the Pattern Report Card system. This management system is called the *PatteRn Evaluation Management and Execution System* (PREMES)[2] framework.

(4) Finally, we present an example execution of the PREMES framework through evaluation of the *Information Secrecy* pattern from Braga et al. [1998].

Metrics are used to measure the quality of the pattern presented to determine and highlight any deficiencies during its creation. With such metrics any improvements made to the pattern can be highlighted and also tracked. Thus providing a repoducible process. Through provision of a reproducible and measured framework, the quality of software design patterns being produced will ultimately improve.

1.2 Organisation

The approach supporting the PREMES framework is described in Section 3. Section 4 introduces the quality indicators upon which the evaluation process is based. Pattern Report Cards and PREMES are described in Sections 5 & 6. An exemplar illustrating how the *Information Secrecy* pattern from Braga et al. [1998] is evaluated using the framework is given in Section 7. We conclude with a discussion over the presented work in Section 8.

2. PROBLEMS WITH PATTERN EVALUATION

By definition, a design pattern is a well-documented recurrent problem within a particular context, paired with a well-documented solution such that the solution is a known good solution [Alexander et al. 1977]. Design patterns are supposed to be solutions to problems; are supposed to be tried and tested; and are supposed to be well documented. When looking to evaluate a pattern these three areas must be considered. Further, to promote a better evaluation practice, pattern evaluation must be: reproducible; consistent; and allow for fine-grained assessment of the presented pattern. Current evaluation systems do not provide comprehensive guarantees in *all* these areas.

The *Shepherding* process, as outlined in Harrison [1999], is a known process for pattern evaluation. However, this process is too generic and relies on pattern authors creating patterns according to a specific style. For instance,

[1] The evaluation of pattern languages is not treated by this framework.

[2] Pronounced /ˈpɹɛ.mis/.

the process does not provide detailed criteria nor guidance over what good and bad patterns are. Nor does the process detail how the solution presented should be evaluated to determine its quality. Further, it makes a tacit assumption that the pattern being evaluated was written according to the advice given in Mezaros and Doble [1997]. This advice tacitly provides a means to ensure good quality patterns by guiding the authors to writing patterns in a good style.

Other existing pattern writing guides such as Wellhausen and Fießer [2011] and Harrison [2004] also detail how notions of quality are accounted for during the pattern writing process. The shepherding process needs enhancement to allow for an evaluation process that is not only reproducible and agnostic to the pattern being written, but also provides detailed evaluation of the presented pattern.

Heyman et al. [2007] details two methods of evaluation. The first determines if the presented pattern is a pattern, and the second provides assessment over documentation quality. This approach does not evaluate the pattern in other areas. The approach of Halkidis et al. [2004] is purely for security patterns and concentrates on assessment of solution quality and not documentation. Laverdière et al. [2006] presents a comprehensive set of criteria for security design pattern evaluation using the *Six Sigma* approach. However, no implementation criteria nor how to assess patterns using this approach was given.

One of the difficulties in constructing an evaluation system for software design patterns is that the subject domain covered is heterogeneous. A single general purpose evaluation system cannot be, and should not be specified. For example, Bunke et al. [2011] detail that, among other things, not all patterns adhere to common pattern templates. Any suggested evaluation process must be tailored to the patterns being evaluated.

A question that naturally arises is: Can a holistic approach be taken to: (a) construct a pattern evaluation system that tests for pattern quality; (b) determines if the pattern presents a solution to a problem, is tried and tested, and well documented; and (c) tailored to the patterns being evaluated?

3. APPROACH

Our approach to pattern evaluation is to establish a series of quality indicators drawn from existing literature that, when tested for, provides an indication over pattern quality. With these indicators, a mixture of qualitative and quantitative evaluation techniques are used to construct an evaluation system that determines how well the indicator has been satisfied. Use of quantitative and qualitative metrics allows for the quality of a pattern to be measured, and for problem areas relating to pattern quality to be highlighted during the engineering process, and thus resolved before publication. A measured approach aids in reducing the subjectivity of the auditor during the evaluation process through the provision of a systemised approach together with summative feedback.

However, provision of summative feedback prohibits for detailed comments to be given over why a pattern fails. This lack of detail can lead to confusion over why a certain grade was allowed when the report card is analysed by the pattern writer. Even more so this can damage the transparency of the evaluation process. Evaluation results can be enhanced by provision of formative feedback as well.

Further, different patterns will have different requirements for their assessment. The *Plan-Do-Check-Act* (PDCA) management cycle is a standardised technique used for establishing bespoke evaluation procedures. Specification of a PDCA management system allows for the establishment and execution of an evaluation system to be tailored to the pattern being evaluated. This system can also be used to provide guidance not only over how best feedback can be given to pattern writers, but also guidance over how to execute the evaluation process.

With this combined framework (management system, evaluation system, and indicators) comes a reproducible means to evaluate patterns. More through the provision of a tailored approach, pattern quality can be assessed at a fine-grained level.

4. QUALITY INDICATORS FOR PATTERNS

Design patterns are represented as documents using a mixture of natural language descriptions and formal models to document the solution's components and their interaction. Common to many patterns are a series of core areas that must be addressed in the document formulation. These are: (a) the *context* in which the pattern is being applied; (b) the *problem* the pattern is solving; (c) the *forces* that drive the choice of solution from the problem; (d) the *solution* presented by the pattern including the solution's dynamics and structure; (e) the *resulting context* from application of the solution; (f) the *relations* with other patterns; and finally (g) *guidance* for pattern application.

When evaluating patterns this document and its contents must inform the evaluation criteria used. Laverdière et al. [2006] has already presented a set of desirable properties for design patterns and nominal measures of their quality. We take these properties and other existing bodies of work on pattern evaluation [Wellhausen and Fießer 2011; Harrison 2004; Bunke et al. 2011; Mezaros and Doble 1997; Heyman 2013] and present a set of quality indicators that, if tested for, can be used to determine the quality of the presented pattern. The quality indicators are presented below and have been grouped according to the three notions that a design pattern is *supposed* to be: well-documented; tried & tested; and solutions to problems.

4.1 Quality of the Pattern Presented

Research has identified that a proportion of the presented patterns fail at being patterns [Bunke et al. 2011; Heyman et al. 2007; Winn and Calder 2002]. Design patterns are supposed to be *solutions to problems.* But what exactly is a design pattern supposed to be? Attempts at providing more formal definitions do exist [da Silva Júnior et al. 2013; Dietrich and Elgar 2005; Shiroma et al. 2010; Bayley and Zhu 2010; Dong et al. 2007a; Dong et al. 2007b; Dong et al. 2004], however, these definitions view design patterns as software artefacts and do not necessarily take into account a pattern's emergent properties of documentation quality, and goodness of solution. The first set of indicators determine if the presented pattern is actually a pattern, and address in part the core areas of: *context*, *problem*, *forces*, and *solution*.

Definition 4.1 (*Pattern Coherency*). The pattern presented must present a solution to a recurrent problem for a particular context. The presented solution should match the level of abstraction of the presented problem (its forces), and also match the context in which the problem exists.

Definition 4.2 (*Pattern Atomicity*). The pattern presented must be an entity from which other sub-patterns cannot be extracted. A single pattern should present a single discrete problem; the problem should not be composed of multiple problems. Related problems should be referenced in the pattern document, and a group of linked patterns should be presented as a pattern language.

Definition 4.3 (*Problem Independence*). The problem being described, together with its forces should not be influenced in description and construction by the presented solution. The problem should be independent to any presented solution.

4.2 Quality of Solution Presented

Design patterns are supposed to be *tried & tested* solutions. Many pattern documents and guides stress that a design pattern is the successful resolution of a series of problem forces by a series of actions within a particular context [Wellhausen and Fießer 2011]. When presented with a pattern the solution presented is supposedly a good solution to the presented problem. These next set of quality indicators are concerned with the solution quality, and address in part the core areas of: *solution*, and *resulting context*.

Definition 4.4 (*Solution Appropriateness*). The solution presented should be a solution for the presented problem. The presented solution should address the described problem in its entirety for the given context, and not present a general solution that is applicable to other problems.

Definition 4.5 (*Solution Complexity*). The presented solution should not be overly complex, and should not be difficult to apply. Solution complexity will effect on the applicability of the pattern and how well it can be deployed to address the presented problem.

Definition 4.6 (*Solution Effectiveness*). The pattern document should provide evidence of the solution's effectiveness and robustness for addressing the presented problem. Such evidence can be used determine the quality of the solution and how well the problem is addressed.

4.3 Quality of Pattern Presentation

The final set of indicators are related to pattern presentation. Design patterns are supposed to be *well-documented*. Poorly presented ideas will be poorly received by readers. As design patterns are ostensibly used for domain knowledge transfer (from domain expert to non-domain experts) the quality of the pattern document should also be assessed. Attention to presentation and content was identified in Harrison [2004], Heyman et al. [2007], Yoshioka et al. [2008], and Wellhausen and Fießer [2011]. This final set of indicators provides treatment at the documentation level of all areas of concern within the pattern.

Definition 4.7 (*Pattern Structure*). Pattern templates are used to provide a common structure for describing like patterns. A coherent structure provides better presentation of the topics. Measuring the quality of adherence to a known template can be used to indicate good structure. Further the chosen template should be suitable for the presented pattern.

Definition 4.8 (*Pattern Legibility*). The language used in the pattern documentation should convey clearly to the target audience the ideas being described. Use of overly complex language, or too simple a use of language, can hamper the reader's ability to comprehend the presented material.

Definition 4.9 (*Presentation Accessibility*). Whereas the previous indicator concentrated on the writing quality this indicator looks at other aspects of the presentation. The problem, solution, and ideas should be presented in a way that promotes accessibility and does not hamper the readers ability to familiarise themselves with the concepts. Terminology, and concepts should be explained clearly and presented appropriately.

5. PATTERN REPORT CARDS

This section presents *Pattern Report Cards*, an evaluation system that can be used to test against the described indicators from Section 4. This system took inspiration from educational report cards that detail how well a student is performing. For each of the three quality areas several qualitative and quantitative techniques are used to gauge the pattern's quality. These techniques are used to create data points that can be used to track pattern quality during the engineering process. Further, the assessment process has been designed to present pattern writers and auditors with a repeatable means to determine pattern quality, and indicate areas of improvement. The remainder of this section details the techniques used, and how they can indicate pattern quality.

5.1 Quality of Pattern Presented

The first part of the report card grades the pattern according to pattern quality using qualitative measurements. For each of the quality indicators we ask auditors to grade the pattern using the provided grading schemes. The schemes have been designed such that better grades indicate patterns that have the correct form, and lesser grades degradation of said form.

5.1.1 *Coherency Grade.* The first grade relates to pattern coherency, and has the following grading scheme:

A	The pattern presents a well defined problem that is recurrent, describes a solution for that problem in a particular context, and both problem and solution are at the same level of abstraction.
B	The pattern presents a reasonably defined problem that is recurrent, the solution addresses most of the problem presented for a particular context, and the level of abstractions for the solution and problem are the same.

C	The pattern presents an ill-defined problem that is somewhat recurrent, the solution addresses a substantial portion of the presented problem for a particular context, and the level of abstraction for the solution and problem is similar.
D	The pattern presents an ill-defined but not recurrent problem, the solution only addresses part of the problem specified for a particular context, and the problem and solution have similar yet differing levels of abstraction.
E	The pattern does not present a well defined nor recurrent problem, the solution does not address the problem for any context, and the problem and solution have differing levels of abstraction.

5.1.2 *Atomicity Grade.* The second grade relates to pattern atomicity, and has the following grading scheme:

A	The pattern is sufficiently constrained and cannot be decomposed into smaller patterns.
B	The pattern presented is suitably constrained and aspects of the pattern could be turned into other patterns.
C	The pattern presented is not constrained to a single problem and aspects should be decomposed into other patterns.
D	The pattern presented addresses several unrelated problems and should be decomposed into several smaller patterns.
E	The pattern presented is in fact a pattern language and addresses to many inter-related problems, and thus can be decomposed into smaller patterns.

5.1.3 *Problem Independence.* The final grade for pattern quality relates to the independence of the problem presented.

A	The problem presented is independent of the solution presented, and the problem forces are not indicative of the solution being proposed.
B	The problem presented is not influenced by the solution presented, and the problem forces bear some resemblance to the issues affecting the solution.
C	The problem presented is influenced somewhat by the solution presented, and the problem forces resemble issues affecting the use of the solution.
D	The problem presented is influenced by the solution presented, and the problem forces are descriptive of issues affecting the solution.
E	The problem presented is directly influenced by the solution presented, and the problem forces explicitly describe problems affecting the solution and not the problem.

5.2 Quality of Solution Presented.

The next section of the report card determines solution quality according to the problem being presented. For this next set of grades quantitative values are obtained primarily from qualitative measurements. For each indicator of solution quality a different measurement and transformation is presented. Unlike the previous grading section, the formulations and values presented are not generic to all patterns and will differ on a per pattern basis. Where appropriate suitable descriptions are provided.

5.2.1 *Solution Appropriateness.* The first grading is for solution appropriateness. Appropriateness is made quantifiable by establishing a metric made from qualitative values. To calculate this metric, one must first identify the problem forces, and assign a weighting (percentage) to indicate the importance of each force within the problem description. Secondly, a grading scheme[3] is defined to grade how well the solution addresses each of the problem forces in turn. This grading scheme must also be representable by a scalar number. For each force, multiplying each of the resulting grades by the force's weighting a metric for solution appropriateness, namely the *weighted solution satisfaction* can be calculated.

Definition 5.1 (*Weighted Solution Satisfaction*). Given a set of forces $\mathcal{F} = \{f_0, \ldots, f_n\}$. Let $\mathcal{W} = \{w_0, \ldots, w_n\}$ be a set of weightings for each $f \in \mathcal{F}$ such that $\sum_{i=0}^{n} w_i = 100$. Let $\mathcal{G} = \{1, 2 \ldots, n\}$ be a bounded range of integer values that represents a grading scale. Let $\mathcal{E} = \{e_0, \ldots, e_n\}, e_i \in \mathcal{G}$ be a set of evaluation values for each $f \in \mathcal{F}$. The weighted solution satisfaction for a pattern is calculated as follows:

$$\sum_{i=0}^{n} w_i \times e_i$$

5.2.2 *Solution Complexity.* The second grading is related to solution complexity. Within Design Pattern literature, a solution's structure and dynamics are often modelling using the *Unified Modelling Language* (UML). The UML standard provides a series of language agnostic modelling languages that can model separate aspects of the presented solution. When presented with UML descriptions for a solution, the solution's complexity can be inferred by calculating the complexity of the presented models. There are well known metrics for UML Class Models [Yi et al. 2004; Mahmood and Lai 2005; Marchesi 1998; Manso et al. 2003], however, metrics for other UML models such as deployment, component, and sequence are not so well developed. For code based patterns, metrics for code quality can also be constructed [Le Goues and Weimer 2012]. When determining the complexity of the given solution a set of metrics for the models being presented must also be decided upon. For patterns that do not provide models the auditors must detail a grading scheme that allows for the solutions presented to be graded according to their complexity. For example:

Complex	The solution is too complex with a structure that contains too many modules that have to many relations. Further, the interactions between the components are too many for the interactions described.
Adequate	The solution presented has a structure and set of dynamics that are suitable for addressing the problem presented.
Simple	The solution presented has a structure and set of dynamics that are too simplistic for the problem being addressed. This solution does not capture enough detail for the problem presented.

5.2.3 *Solution Effectiveness.* The effectiveness of a presented solution determines pattern quality according to how well the problem is addressed. Whereas complexity can be quantitatively measured, effectiveness is more of a qualitative measurement performed through walk-throughs with especial regard to both normal and abnormal usage. Some quantitative measures can come from taking the pattern requirements and generating test cases to

[3]The grading scheme defined for force satisfaction can also be defined to indicate over-engineered solutions.

apply to the pattern design. Guidance from Requirements Engineering evaluation practices will advise this process with especial regard to the interface, inputs and outputs.

Naïvly, a simple solution is to look for evidence of metrics and evaluation criteria with the presented pattern document. Resulting in a naïve grading scheme that can be employed to judge solution effectiveness:

A	The pattern presents ample evidence that the solution presented is effective.
B	The pattern presents enough evidence that the solution presented is effective but some aspects of the solutions effectiveness are not described.
C	The pattern provides links to evidence that the presented solution is effective in addressing the problem.
D	The pattern alludes to the effectiveness of the solution but does not categorically present evidence attesting to the fact.
E	The pattern does not present any evidence that the presented solution is effective in addressing the problem.

5.3 Quality of Pattern Presentation.

The final set of grades are for pattern presentation. Here the grading schemes presented are a mixture of quantitative values constructed from qualitative measurements, and qualitative grading. Pattern presentation can be assessed according to: adherence to known templates; use of representational aides; and quality of writing used.

5.3.1 *Pattern Structure.* Heyman et al. [2007] presented a methodology for assessing the pattern documentation quality according to how well a pattern adheres to a given template: *Weighted Adherence to Pattern Template*. This indicator provides a mark for the adherence a given pattern has towards a specified template. Each heading in a given pattern template is associated with a weighting indicating the importance of each heading within the template. When analysing a pattern each heading is graded to indicate implementation quality for each element. An adherence metric can then be calculated through summation of the scores for each element. The higher the score the greater the adherence to the template.

Definition 5.2 (*Weighted Adherence to Pattern Template*). Given a pattern template $\mathcal{T}$. Let $\mathcal{W} = \{w_0, \ldots, w_n\}$ be a set of weightings for each $t \in \mathcal{T}$ such that $\sum_{i=0}^{n} w_i = 100$. Let $\mathcal{G} = \{1, 2 \ldots, n\}$ be a bounded range of integer values that represents a grading scale. Let $\mathcal{E} = \{e_0, \ldots, e_n\} where e_i \in \mathcal{G}$ be a set of evaluation values for each $t \in \mathcal{T}$. The weighted adherence to a pattern templates is calculated as:

$$\sum_{i=0}^{n} w_i \times e_i$$

5.3.2 *Pattern Legibility.* The legibility of a pattern document can be assessed using existing (and standard) readability metrics such as Flesch-Kincaid, Coleman, and FOG [Kincaid et al. 1975]. In readability metrics low scores will represent the use of simplified language constructs, and higher scores represent more complex. Often such readability metrics are interpreted according to the American Grade Level to allow easy interpretation of the result. These metrics can be used to indicate how advanced, or simplified, the language used will be.

5.3.3 *Presentation Accessibility.* The final grading scheme is presented for assessing pattern accessibility. Accessibility can be determined by judging the pattern according to the terminology used and clarity of the descriptions. This grading schemes requires that the existence and suitability of additional presentation attributes be assessed together with the clarity of presentation. Examples of additional presentation attributes can include the existence of, for example UML models, diagrams, references to existing work and usage, and code examples.

A	The pattern is presented in an accessible manner using clear language. The concepts and terminology used is explained appropriately.
B	The pattern is presented in an accessible manner but does not use clear language. The concepts and terminology are explained using unclear language.
C	The pattern is presented using clear language but is nonetheless inaccessible. The use of terminology is given but poorly explained and presented unsatisfactorily.
D	The pattern is presented using unclear language, makes use of terminology that is poorly explained, and presented unsatisfactorily.
E	The pattern is presented using unclear and inaccessible language. Terminology used is unfamiliar and unclear to the reader. The pattern is presented poorly.

6. THE PREMES FRAMEWORK

The previous section introduced Pattern Report Cards. Several of the schemes described are not generic to all patterns and must be tailored per pattern. The PREMES approach uses the PDCA cycle (Section 3) to manage the execution of the evaluation process and to introduce the required tailoring. This section describes the activities required for each of the four stages, and the rationale behind each stage where appropriate. The description of PREMES has been given under the assumption that the pattern is to be reviewed by a team of auditors.

6.1 The Planning Stage

The first stage is preparatory and requires auditors to establish the scope and extent of the evaluation. The required activities involved are:

(1) Identification of the patterns that are to be evaluated.
(2) Agreement on the weightings to be used in the Report Card Process.
(3) Identification of the Pattern Template used and weightings decided upon for each of the headings.
(4) Agreement on the readability metric used for analysing language style.
(5) Agreement on marking criteria for each of the qualitative grading schemes.
(6) Agreement on number of iterations that the cycle will go through.
(7) Agreement on how to collate the results of the different report cards into a single card.
(8) Agreement on how results are to be reported.

This stage explicitly ensures that there is a consensus for both how the evaluation is to be conducted and how each aspect of the evaluation is to be performed. This will included identifying how the evaluation is to be tailored for the particular set of patterns presented. Of note is the agreement on how results are to be reported, allowing auditors to decided precisely what results the pattern writer will receive. For example, provision of free flow formative feedback alongside the summative grading.

6.2 Grading the Pattern

The second stage requires the execution of the pattern report card process by the pattern auditors. The required activities involved for each auditor is to:

(1) Grade the Quality of Pattern Presented.
(2) Grade the Quality of Solution Presented.
(3) Grade the Quality of Pattern Presentation.
(4) Creation of the Pattern Report Card.
(5) Detail any formative feedback.

6.3 Analysing the Results

The third stage is results collection and collates the results from each report card into a single one, together with identification of actionable items. The required activities involved are:

(1) Discussion of the results of the report card.
(2) Collation of the results into a single report card according to the agreed upon scheme.
(3) Identification of actionable items, and recommendations for change, from the report card.
(4) Ordering of actionable items in order of precedence.
(5) Creation of a report detailing the results of the Report Card together with formative feedback. The report created must provide a summary of the grading schemes utilised in the evaluation.

This stage provides an agreed upon analysis of the results and present the pattern writer with a coherent set of prioritised actionable items. These items can be produced alongside a single document that contains a single report card for each pattern and more formative textual feedback.

6.4 Results Reporting

The final stage of the process requires the reporting of the results back to the pattern writers, improvement of the pattern according to recommendations, and re-submission of the pattern for re-evaluation. The required activities involved are:

(1) Submission of the report card and recommendations to the pattern writer.
(2) The pattern writer will make changes based upon the recommendations.
(3) The pattern writer will submit the pattern for review.
(4) The pattern auditors will reevaluate the pattern accordingly.

7. EVALUATING THE INFORMATION SECRECY PATTERN

This section details how the PREMES framework can be used to evaluate the INFORMATION SECRECY pattern from Braga et al. [1998]. This section will not explicitly report all aspects of the evaluation but concentrate on reporting how the framework was used to conduct the evaluation.

7.1 The Planning Stage

For analysing the pattern the following decisions were made:

Quality of Pattern Presented: For this pattern, the first part of the evaluation does not require decisions to be made before execution of the evaluation process.

Quality of Solution Presented: To calculate the *weighted solution satisfaction* metric from Section 5.2.1 each force shall be treated equally and given the same weighting (i.e. $\mathcal{W} = \{33.33, 33.33, 33.33\}$) with a grading scheme of $\mathcal{G} = \{0, 1, 2\}$ used. These evaluation values were taken from Heyman et al. [2007]. For assessment of solution complexity, the grading scheme presented in Section 5.2.2 will be used. Assessment of solution effectiveness shall be from a naïve assessment that looks for evidence of effectiveness.

Quality of Pattern Presentation: For calculating the weighted adherence to a pattern template (Section 5.3.1) it was noted that the templates used in Braga et al. [1998] consisted of six headings that are all equally important. Thus, the weightings are for each heading shall be 16.67%. The grading scheme used will be the same one used in Heyman et al. [2007] i.e. $\mathcal{G} = \{0, 1, 2\}$. Pattern Legibility shall be assessed using the Flesch-Kincaid readability metric.

7.2 Grading the Pattern

The summative scores with salient formative feedback for each of the quality indicators are:

Indicator	Grade	Comments
Coherency Grade	E	The problem is not defined sufficiently enough to warrent comparison to the solution.
Atomcity Grade	B	The pattern is well contained but some aspects such as algorithm selection could be treated using the pattern format.
Problem Independence	E	The pattern presented details forces that are too tied to the solution, namely how to use cryptography, and the forces are not solution independent.
Solution Appropriateness	0	The forces are not addressed by the solution at all.
Solution Complexity	Adequate	The structure and dynamics of the solution are adequate for the problem of cryptography and do not present a complex solution from a design view point.
Solution Effectiveness	D	The pattern links to evidence of the solution in use but does not link to evidence of its effectiveness in providing information secrecy.
Pattern Structure	50%	Each of the presented section were provided minimally, and requires more information to be come satisfactorily provided.
Pattern Legibility	Grade Level 8	The reading level was presented at grade level 8, this implies that it is understandable for students aged between 13–15. This implies a good level of writing.
Presentation Accessibility	C	Although the pattern was presented using clear language key aspects of the problem and solution were not explained at all and required prior knowledge to be known for example algorithm selection.

7.3 Analysing the Results

The grades and scores in Section 7.2 can be used to produce a report that provides detailed formative feedback on the deficiencies presented in the pattern along side the already reported summative feedback. For example, the quality of the pattern can be summarised as follows:

> *From the grades presented the pattern fails to be a pattern. It does not present an appropriate solution that addresses the described forces, and does not detail the effectiveness of the presented solution. Further, although the pattern is presented clearly, it is not accessible as key information surrounding the solution and its application are missing or not elaborated on.*

The presented report can be concluded with a set of prioritised actionable items that provides advice on which areas are required for improvement. A preliminary set of actionable items includes but not limited to: further elaboration of the problem with a set of forces that is solution independent; more information surrounding the solution and its effectiveness; and links to information pertaining to algorithm selection.

7.4 Results Reporting

For this stage the results document from the analysis stage will be sent to the pattern author.

8. DISCUSSION

The PREMES framework has coalesced into a single solution various techniques and methodologies from various different fields such as Requirements Engineering, Software Design, Testing, and Modelling. The resulting solution evaluates patterns using a mixture of qualitative evaluation, and metrics collected from quantitative and qualitative data sources. Use of this iterative process to track feedback allows auditors a means to determine the effort placed in writing the pattern as well as the ability to track deficiencies. This section discusses the proposed framework according to scope, style of feed back, measurement techniques, and areas for improvement.

8.1 Scope of Evaluation

Section 4 introduced the quality indicators for patterns. A question naturally arises over how complete these indicators are in determining evaluation quality.

An aspect not explicitly mentioned, nor tested for, is that of pattern usability. How usable is the pattern document by non-domain experts? Pattern usability can be used to identify how accessible the pattern document is, and ease of pattern application. For assessment of pattern usability such testing would require the use of user studies. Thimthong et al. [2013] has explored the use of user studies for pattern evaluation. However, use of user studies should be limited as the usefulness of such studies can be ineffective and unhelpful if done improperly [Greenberg and Buxton 2008]. Nonetheless, if we look at the quality indicators and the tests for those indicators, important aspects of usability are infact tested for. For example, accessibility of the language used to introduce the concepts, the known and perceived complexity of the solution presented, and how well the auditor believes the concepts are made accessible.

A secondary problem in determining the scope of the analysis is that most of the report card system is left purposely undefined. It is up to the pattern auditor's themselves to determine precisely what is involved in the analysis. For the quantitative portions of the analysis such concerns can be minimised through specification of approved analysis techniques whose scope and limitations are known. For the qualitative measurements issues, more guidance can be produced to guide the auditor in producing the qualitative value.

Several of the indicators use the information presented in the pattern as testing criteria. If this information is badly presented this will affect the quality of the resulting evaluation. This problem is addressed as part of the evaluation process. Having multiple rounds of evaluation can allow pattern auditors a means to ensure that the information is sufficiently presented before the evaluation process.

8.2 Evaluation of Pattern Languages

Pattern languages are sets of interconnected patterns that when combined provide a coherent solution to a problem that cannot be described using a single pattern. The combined use of several patterns together may affect the quality of the solution being presented to address the larger problem being tackled. Pattern language quality is just as important as individual pattern quality. However, the PREMES framework evaluates patterns in isolation, and does not take into account related patterns, nor the evaluation of pattern languages. The PREMES approach is limited in effectiveness when used against pattern languages. Future work will be to investigate if the approach can be extended to include pattern language evaluation.

8.3 Formative vs Summative Feedback

When using Pattern Report Cards in isolation the feedback is summative and prohibits further explanation of why grades where awarded. When combined with the management process more formative feeback can be given alongside the report card, allowing explicit mention to be given to a pattern's deficiencies. More so, during the checking phase (Section 7.3) there are no restrictions on the style and detail of the reports created.

8.4 Qualitative or Quantitative

The evaluation techniques proposed use quantitative analysis, quantitative analysis derived from qualitative measurement, and pure qualitative analysis. These techniques allow for the pattern quality to be measured, and also made reproducible However, such a mixture of evaluation techniques, especially the use of qualitative measurements, increases the difficulty in ensuring consistent reproducible evaluation scoring. Pure qualitative measurement is known for being highly subjective and open to interpretation, with quantitative scoring based on qualitative measurement less so. The requirement of subjective evaluation could have a detrimental effect on the quality of analysis. In its current state Pattern Report Cards offers too high a degree of subjectivity in its evaluation with too many of the grading schemes being purely qualitative. Future work will be to investigate how these subjective aspects of the evaluation can be made more objective or the subjectivity cancelled out. There are several approaches under consideration.

Better Guidance A naïve first approach is to produce more guidance for each of the grading schemes presented. This guidance would provide a more detailed description over what a pattern would look like if it was to be awarded a specific grade. This will facilitate use of the grading scheme in a more effective manner.

More Quantitative from Qualitative There will always be a subjective aspect relating to pattern quality if qualitative techniques are employed. A second more practical approach is to reduce the size of the qualitative measurements being taken. This will reduce the affect that an auditors subjectivity will have on the grading. This can be achieved by transforming the purely qualitative measurements into a quantitative grade calculated from qualitative measurements. Each of the quality indicators can be broken down into individual attributes that can be measured qualitatively and these values used to construct a quantitative grade. This is the approach taken for weighted adherence to pattern template.

Psychometric Questionnaires A third and final approach is to fully embrace and acknowledge the existence of subjectivity in quality evaluations. With this, psychometric testing techniques such as Likert and Guttman Scales, could be used to determine the auditor's attitude towards pattern quality using these techniques. In particular Likert Scales measure a subject's response in terms of their agreement or disagreement. When applied to patterns, a psychometric test can be devised to determine the auditor's agreement level over the quality of the presented pattern. Key to the use these techniques is the need to minimise the inherent biases within the subject's own response such that a true picture of the subject's attitude is measured. This can be used to control and minimise the inherent bias as presented by the auditor.

8.5 Need for an Overall Grade

Pattern Report Cards does not provide an overall grade for the pattern being evaluated. Although the individual grading provides a detailed assessment over how well a pattern performed, the lack of an overall grade may trouble the pattern writer over how good their presented pattern is. This raises a secondary question of: *Can differing levels of quality be established given the disparate set of grading tools used for each quality indicator?* To aid in the creation of an overall grade, the disparate set of grades needs to be interpreted to provide a single value that is indicative of pattern quality. That is, a grade conversion and collation algorithm needs to be created. Given the second approach to increasing objectivity in the evaluation given in Section 8.4, a better approach may be to harmonise the grading schemes such that the same reporting scale is used. Each indicator would be divided into smaller weighted attributes that are used to calculate a quality value for the indicator using the same reporting scale. The indicators can also be assigned a weighting (denoting importance) such that a weighted average can be constructed from the grades presented. From this differing levels of quality could be established based on the possible range of the final value. This is left as future work.

9. CONCLUSION

There is much literature on design pattern evaluation. However, when looked at individually the work presented does not present a holistic treatment of patterns during the evaluation process. Building on top of existing work the PREMES framework is a managed process for pattern evaluation that uses a PDCA cycle. This cycle contains clear stages in which the auditors are to first: design and plan their evaluation process; evaluate the presented patterns; check the results; and feed the reporting back to the pattern writer. At the heart of this framework are *Pattern Report Cards.* These report cards assess a pattern according to a set of quality indicators. How the pattern is graded can be tailored per pattern and the PDCA cycle makes this tailoring an explicit process. Pattern Report Cards presents a measurable and reproducible evaluation technique for design patterns, that allows problem areas of a pattern to be highlighted during evaluation.

The PREMES framework and Pattern Report Card system is still in its infancy and more work needs to be performed to better hone the process. Future areas of work that have been identified are as follows:

Metrics for Pattern Evaluation Several metrics for patterns have been introduced in this paper. Future work will be to investigate if more quantitative-based metrics for patterns can be established. These metrics can then be fed into the evaluation process for more informative and quantified grading of patterns.

Objective Evaluations Evaluation is an inherently subjective process. Such a subjective process can be made more objective through the employment of quantitative (through qualitative measurements) grading. However, unless the grading is purely quantitative then a degree of subjectivity will be present in the evaluation. Future work will be to investigate how bias in the evaluation can be mitigated.

Reproducible Evaluations Several of the evaluation techniques imposed within pattern report cards can be made automatable, for instance calculating the presentation metrics. Future work will be to investigate to what degree that the report card process can be made more automatable, especially with more quantitative grading based on qualitative measurement.

Proving the Evaluation Process The usability of the evaluation process, and the quality of evaluation arising from those not familiar with the process, has yet to be examined. Future work will be to examine the application of the process by pattern auditors and writers, and test the process on a wide range of patterns.

Results Reporting Pattern Report Cards presents pattern auditors with a wealth of information. An area of future work will be to investigate how the results of the report card can be reported 'best' to pattern auditors and pattern writers.

Evaluating Pattern Languages The PREMES framework evaluates patterns in isolation, and does not take into account related patterns, nor has it been design to evaluate pattern languages. Future work will be to investigate how the framework can be extended to evaluate pattern languages.

REFERENCES

Christopher Alexander, Sara Ishikawa, Murray Silverstein, Max Jacobson, Ingrid F. King, and Shlomo Angel. 1977. *A Pattern Language: Towns, Buildings, Construction.* Oxford University Press, New York.

Ian Bayley and Hong Zhu. 2010. Formal specification of the variants and behavioural features of design patterns. *Journal of Systems and Software* 83, 2 (2010), 209–221. DOI:http://dx.doi.org/10.1016/j.jss.2009.09.039

Alexandre M. Braga, Cecilia M. F. Rubira, and Ricardo Dahab. 1998. Tropyc: A Pattern Language for Cryptographic Software. In *Pattern Languages of Programs PLoP 1998.* http://www.dcc.unicamp.br/ic-tr-ftp/1999/99-03.ps.gz

Michaela Bunke, Rainer Koschke, and Karsten Sohr. 2011. Application-Domain Classification for Security Patterns. In *PATTERNS 2011, The Third International Conferences on Pervasive Patterns and Applications*. ThinkMind, 138–143.

Luis Sérgio da Silva Júnior, Yaan-Gael Guéhéneuc, and John Mullins. 2013. *An Approach to Formalise Security Patterns*. Technical Report. École Polytechnique de Montréal, Montréal Quebéc.

Nelly A. Delessy and Eduardo B. Fernández. 2008. A pattern-driven security process for SOA applications. In *Proceedings of the 2008 ACM symposium on Applied computing (SAC '08)*. ACM, New York, NY, USA, 2226–2227. DOI:http://dx.doi.org/10.1145/1363686.1364217

J. Dietrich and C. Elgar. 2005. A formal description of design patterns using OWL. In *Software Engineering Conference, 2005. Proceedings. 2005 Australian*. 243–250. DOI:http://dx.doi.org/10.1109/ASWEC.2005.6

Jing Dong, Paulo S.C. Alencar, and Donald D. Cowan. 2004. A behavioral analysis and verification approach to pattern-based design composition. *Software and Systems Modeling* 3 (2004), 262–272. Issue 4. DOI:http://dx.doi.org/10.1007/s10270-004-0056-z

Jing Dong, Paulo S. C. Alencar, Donald D. Cowan, and Sheng Yang. 2007a. Composing pattern-based components and verifying correctness. *J. Syst. Softw.* 80, 11 (Nov. 2007), 1755–1769. DOI:http://dx.doi.org/10.1016/j.jss.2007.03.005

Jing Dong, Tu Peng, and Yajing Zhao. 2007b. Model Checking Security Pattern Compositions. *Quality Software, International Conference on* 0 (2007), 80–89. DOI:http://dx.doi.org/10.1109/QSIC.2007.37

Saul Greenberg and Bill Buxton. 2008. Usability Evaluation Considered Harmful (Some of the Time). In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08)*. ACM, New York, NY, USA, 111–120. DOI:http://dx.doi.org/10.1145/1357054.1357074

Spyros Halkidis, Alexander Chatzigeorgiou, and George Stephanides. 2004. A Qualitative Evaluation of Security Patterns. In *Information and Communications Security*, Javier Lopez, Sihan Qing, and Eiji Okamoto (Eds.). Lecture Notes in Computer Science, Vol. 3269. Springer Berlin / Heidelberg, 251–259. DOI:http://dx.doi.org/10.1007/978-3-540-30191-2_11

Neil B. Harrison. 1999. The Language of Shepherding: A Pattern Language for Shepherds and Sheep. Online. (1999).

Neil B. Harrison. 2004. Advanced Pattern Writing: Patterns for Experienced Pattern Authors. Online. (2004).

Thomas Heyman. 2013. *A Formal Analysis Technique for Secure Software Architectures (Een formele analysetechniek voor veilige softwarearchitecturen)*. Ph.D. Dissertation. https://lirias.kuleuven.be/handle/123456789/389365

Thomas Heyman, Koen Yskout, Riccardo Scandariato, and Wouter Joosen. 2007. An Analysis of the Security Patterns Landscape. In *Proceedings of the Third International Workshop on Software Engineering for Secure Systems (SESS '07)*. IEEE Computer Society, Washington, DC, USA, 3. DOI:http://dx.doi.org/10.1109/SESS.2007.4

J P Kincaid, Robert P Fishburne, Jr, Richard L Rogers, and Brad S Chissom. 1975. *Derivation of New Readability Formulas (Automated Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy Enlisted Personnel*. Technical Report. 41 pages. Research branch rept. ADA006655.

M.-A. Laverdière, A. Mourad, A. Hanna, and M. Debbabi. 2006. Security Design Patterns: Survey and Evaluation. In *Electrical and Computer Engineering, 2006. CCECE '06. Canadian Conference on*. 1605–1608. DOI:http://dx.doi.org/10.1109/CCECE.2006.277727

C. Le Goues and W. Weimer. 2012. Measuring Code Quality to Improve Specification Mining. *Software Engineering, IEEE Transactions on* 38, 1 (Jan 2012), 175–190. DOI:http://dx.doi.org/10.1109/TSE.2011.5

Sajjad Mahmood and Richard Lai. 2005. Measuring the Complexity of a UML Component Specification. *Quality Software, International Conference on* 0 (2005), 150–160. DOI:http://dx.doi.org/10.1109/QSIC.2005.39

MaEsperanza Manso, Marcela Genero, and Mario Piattini. 2003. No-redundant Metrics for UML Class Diagram Structural Complexity. In *Advanced Information Systems Engineering*, Johann Eder and Michele Missikoff (Eds.). Lecture Notes in Computer Science, Vol. 2681. Springer Berlin Heidelberg, 127–142. DOI:http://dx.doi.org/10.1007/3-540-45017-3_11

M. Marchesi. 1998. OOA metrics for the Unified Modeling Language. In *Software Maintenance and Reengineering, 1998. Proceedings of the Second Euromicro Conference on*. 67–73. DOI:http://dx.doi.org/10.1109/CSMR.1998.665739

Gerard Mezaros and Jim Doble. 1997. Pattern languages of program design 3. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, Chapter A pattern language for pattern writing, 529–574. http://dl.acm.org/citation.cfm?id=273448.273487

Yuki Shiroma, Hironori Washizaki, Yoshiaki Fukazawa, Atsuto Kubo, and Eduardo B. Fernandez. 2010. Model-Driven Application and Validation of Security Patterns. In *Proceedings of the 10th Conference on Pattern Languages of Programs (PLoP '10)*.

Tharis Thimthong, Thippaya Chintakovid, and Soradech Krootjohn. 2013. Evaluating Design Patterns of Commercial Web Applications using Net Easy Score. *I.J. Information Technology and Computer Science* 5, 8 (July 2013). DOI:http://dx.doi.org/10.5815/ijitcs.2013.08.09

Tim Wellhausen and Andreas Fießer. 2011. How to write a pattern? A rough guide for first-time pattern authors. In *Proceedings of the 16th European Conference on Pattern Languages of Programs (EuroPLoP '11)*. ACM, New York, NY, USA, Article 5, 9 pages. DOI:http://dx.doi.org/10.1145/2396716.2396721

T. Winn and P. Calder. 2002. Is this a pattern? *Software, IEEE* 19, 1 (2002), 59–66. DOI:http://dx.doi.org/10.1109/52.976942

Tong Yi, Fangjun Wu, and Chengzhi Gan. 2004. A Comparison of Metrics for UML Class Diagrams. *SIGSOFT Softw. Eng. Notes* 29, 5 (Sept. 2004), 1–6. DOI:http://dx.doi.org/10.1145/1022494.1022523

Nobukazu Yoshioka, Hironori Washizaki, and Katsuhisa Maruyama. 2008. A Survey on Security Patterns. *Progress in Informatics* 5 (2008), 33–47. DOI:http://dx.doi.org/10.2201/NiiPi.2008.5.5

Received May 2015; revised Oct 2015; accepted Feb 2015