A Bottom Up Approach to Model Based Program Validation

Thomas Calder and Yngve Lamo

Bergen University College, Norway
Thomas.Peter.Calder@stud.hib.no & yla@hib.no

Abstract. Validators are used to ensure that internet based services present information in an adequate way to the end users. However, today's validators are not transparent making it difficult for the users to understand their feedback. To enhance the transparency we propose a MDE based approach to program validation, where domain requirements are presented in a metamodel. The validation is based on a bottom up approach where programs are parsed and represented as models before they are checked against the metamodel.

Keywords: Model Driven Engineering, Model Driven Reverse Engineering, Meta-model hierarchies, Model Based Validation

1 Introduction

As the internet becomes an increasingly more important tool for communication, it is necessary to ensure that not only is the quality of internet based digital services adequate, but that they are also equally accessible to all end users. Furthermore, as web based technologies continue to develop rapidly, the need arises for automated tools that can ensure that web pages and digital services comply with the latest recommended standards. These standards are designed for the benefit of the end user, ensuring that the provided digital service functions adequately and is accessible to all end users, especially those with disabilities. Moreover, by following the appropriate guidelines for the digital service, the developer can be certain that the digital service they have designed, is of a decent level of quality. In other words, these standards are beneficial to both those who develop the digital services, and those who use them. One of the better known accessibility standards is the *The Web Content Accessibility Guidelines 2.0 (WCAG 2.0)* [16]. WCAG 2.0 was developed by the *Web Accessibility Initiative (WAI)*, as a means to provide a common set of standards that ensures the contents of a web page is more accessible to end users with disabilities. These disabilities can range from visual and auditory, to learning and other neurological disabilities. WCAG 2.0 also attempts to make the web content more accessible to older individuals with limited experience with web browsers and similar technologies.

It is common practice to determine the quality of a web page with the aid of an automated web evaluation tool. A *web evaluation tool* can be defined

as an application that inspects if a web page satisfies a set of requirements. Any elements contained within the web page that violates one or more of the requirements are flagged as invalid, and included in a report that is presented at the end of the evaluation process. Some of the more advanced evaluation tools, such as Wave [15] and Sortsite5 [13], suggest possible corrections for the invalid element [8]. There exists a number of web evaluation tools that are able to determine if a web page is valid based on a set of requirements. However, many of these tools are limited to parsing a provided web page, and returning a report highlighting elements that have failed to meet the necessary requirements. The actual evaluation process is obscured from the user, which becomes problematic when we consider the fact that the user has to depend upon the tools ability to evaluate the web page correctly [8]. This is referred to as **the transparency problem** in [6].

Furthermore, the lack of transparency in the evaluation process makes it difficult for the user to evaluate the effectiveness and legitimacy of the web evaluation tool. In [3], Brajnik addresses several aspects that must be assessed when evaluating such a tool:

- **The Completeness Problem:** How can the user determine if the tool has done a complete evaluation, i.e. is the tool returning any false negatives?
- **The Correctness Problem:** How can the user determine the correctness of the tool, i.e. is the tool returning any false positives?
- **The Specificity Problem:** How can the user determine the root cause of the errors reported by the tool?

Finally, the lack of consensus in the number of errors reported by existing web evaluation tools indicate a need for clearer and more precise formulation of the requirements in the WCAG 2.0 guidelines [6]. Moreover the requirements should be specified in such a way that they can be understood by individuals who are not necessarily experts within the current domain, but still must have a correct understanding of the requirements. This could be for instance a developer who is creating a digital service that must meet the requirements specified in WCAG 2.0. In other words not only should the requirements be described precisely, they must also communicate their contents correctly and efficiently to all interested parties. This is referred to as the **The Ambiguity Problem** in [6].

To solve these problems we introduce a validation tool based on the bottom up approach to program validation. The approach utilizes methods and techniques from Model Driven Engineering [4] to determine if a web page satisfies the necessary requirements imposed upon it, and in doing so, resolve if the web page is valid or not. Furthermore by specifying a *Domain Specific Modelling Language (DSML)* [9], it is possible to provide a precise formalisation of the necessary requirements. Due to space limitations, this paper focuses on the conceptual ideas of the bottom up approach, interested readers can find more detailed information about the approach and the validation tool in the master thesis of the first author, [6].

The remainder of this paper is organised as follows: In section 2 we give a high level description of the The Bottom-Up Model Validation approach before we

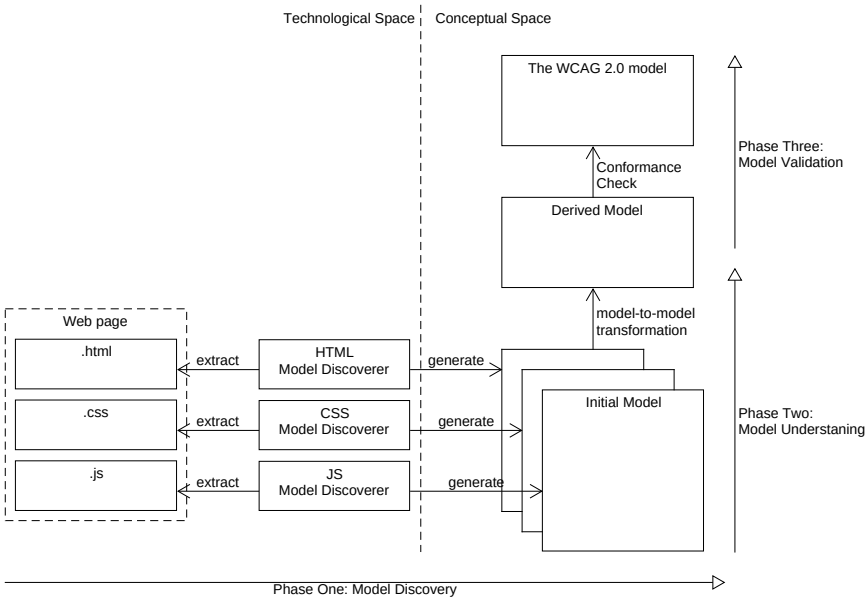

Fig. 1. An overview of the Bottom Up Model Validation approach

give a more detailed description of each of the phases: in section 2.1 we describe the metamodel used in the bottom up validation approach, in section 2.2 we describe the model discovery phase, in section 2.3 the model understanding phase is explained and in section 2.4 is it explained how the model is validated against the WCAG 2.0 standard. In section 3 we discuss how the bottom up approach is related to other contributions before we conclude the paper and envision some possible further work in section 4.

2 Bottom-Up Model Validation

The Bottom-up Model Validation approach, hereby referred to as the *Bottom-up approach*, is inspired by the interactive Bottom-Up Meta-Modelling approach described in [12], where they generate a meta-model which can be used to describe other models from model sketches. In contrast to the Bottom-Up Metamodelling approach the Bottom-up approach generates a model representation of a program artefact and checks if it conforms to an existing meta-model.

Our solution consists of two parts; the first part focuses on designing a DSML capable of providing a clear and precise description of the mandatory requirements. The second part combines the DSML with *Model Driven Reverse Engineering (MDRE)* techniques, to determine if the digital service in question

satisfies all the necessary requirements [6]. Before going in to details, it is important to clarify the two separate spaces the bottom-up approach spans across. The first space consists of technology dependent artefacts such as source code and other platform specific components, this space is referred to as *the technological space*. The second space contains only technology independent components from the specific domain under study, and is referred to as *the conceptual space* [6]. In the Bottom-up approach, the conceptual space consists of graph-based models with diagrammatic constraints. In the context of the problems described in section 1, it can be assumed that the conceptual space is comprehensible to the user, and the technical space is not. This means that by conducting the evaluation process solely in the conceptual space, the bottom up approach provides an adequate solution to *the transparency problem*.

The basic idea behind the Bottom-up approach is to extract an abstract model representation of a web page using Reverse Engineering (RE) combined with MDE. The extracted model can be manipulated, transformed and validated using techniques from MDE. By confirming that the model is valid, our evaluation process can state that the corresponding artefact is also valid. The actual evaluation process itself is reduced to a conformance check between the extracted model representation of the artefact, and the meta-model that models the necessary requirements. Figure 1 presents an overview of the process, and as we can see in the figure, the bottom-up approach consists of three phases:

- **Phase One**: An initial model is extracted from the artefact using platform specific model discoverers. The main goal of this phase is to transfer the problem from the technological space to the conceptual space as early as possible in the validation process, details will be given in section 2.2.
- **Phase Two**: The initial model is manipulated using techniques from MDE. The main goal of this phase is to transform the initial model into a model more suitable for our purposes. The transformed model is referred to as the derived model. More details about this will be presented in section 2.3.
- **Phase Three**: In the third phase, a conformance check between the derived model and the metamodel is executed. The metamodel contains the requirements the artefact must meet. The goal of this phase is to determine if the abstract model representation of the artefact is a valid instance of the metamodel. This will be discussed further in section 2.4.

2.1 Meta-modelling

As previously mentioned, the main purpose of the DSML is to provide a clear and precise definition of the requirements our artefact must satisfy. One of the advantages with describing the requirements using a DSML is that the domain expert get the opportunity to confirm that the requirements have been correctly interpreted by the evaluation process, thus providing the means to reduce the number of false negatives and false positives in the results. The Bottom-Up approach uses a multi level metamodelling hierarchy [1] to represent the requirements our artefact must meet. The top level metamodel describes the

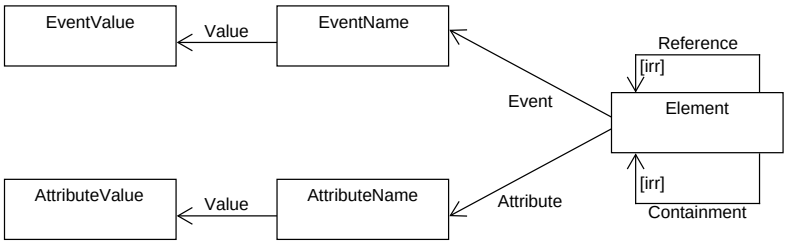

Fig. 2. The HTML Metamodel

types belonging to the domain, and how they are related. Figure 2 shows an example of the top level metamodel describing an element in a web page. The figure shows that each element may have an arbitrary number of attributes and events, and that each element may relate to other elements by either containing them, or by referring to them. The irreflexive constraint, represented using the *[irr]*-notation, prevents any element from containing or referring to itself.

The metamodel at the next level in the metamodelling hierarchy describes the actual requirements we wish to impose upon the artefact. At this level we include in addition to the typing of the artefacts and the relationships between them, modelling constraints that ensure that the corresponding part of the artefact is valid. Figure 3 display a model of an actual requirement from WCAG 2.0. The *1.1.1 Non-text Content requirement* [16] states that all non-textual elements in a web page must have some sort of descriptive text attached to it. Within the web page domain, this requirement is satisfied by ensuring that non-textual content (an image element in this case), has an *alt*-attribute that contains a description of the image. The model in figure 3 enforces this using the *[1..1]*-multiplicity constraint, ensuring that every valid Image element has exactly 1 *alt*-attribute.

The bottom metalevel of the hierarchy is where the derived model is placed after the model understanding phase, this will be elaborated in section 2.4.

2.2 The Model Discovery Phase

The Model Discovery phase is the first part of the MDRE process, a raw model is extracted from the artefact in question, see figure 4. These raw models are

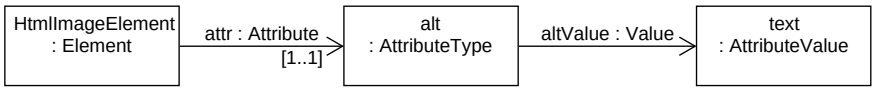

Fig. 3. The Image Element Meta-model

referred to as initial models and are extracted as early as possible during the MDRE process in order to avoid losing any essential information [5]. In order to identify and extract the initial model, platform specific components referred to as *Model Discoverers* are used to populate the initial model with data from the artefact. The Model Discoverer is tailored for a specific technology [5].

We now continue our example involving the *1.1.1 Non-text Content requirement*. At this stage in the bottom-up approach we do not know what parts of the HTML-document are of importance when determining if all non-text content are adequately described. Therefore the initial model is extracted using a HTML specific model discoverer, which transforms the HTML document model to a generic XML based model that conforms the HTML metamodel. This model is of a low level of abstraction ensuring that as little information as possible is lost during the transition from the technological space to the conceptual space. The main purpose of this generated initial model is to serve as a starting point for the model understanding phase [5].

2.3 The Model Understanding Phase

During the Model Understanding phase, chains of model manipulation techniques are used to query and transform the initial model in order to obtain a model of the artefact at a higher level of abstraction, see figure 4. The result of this phase is the Derived Model [5]. The most important function of this phase is to abstract away any unnecessary information contained within the initial model, ensuring that the derived model consists only of relevant content. It is also important that the derived model has been altered in such a way that it suits its intended purpose [5]. Continuing the running example, the HTML image elements contained in the initial model are mapped to an instance of our metamodel. This process is automated using a simple algorithm and set of mapping rules: if there exists a rule for mapping the current HTML element to an element in Metamodel A, the element is included in the Derived Model A, otherwise the

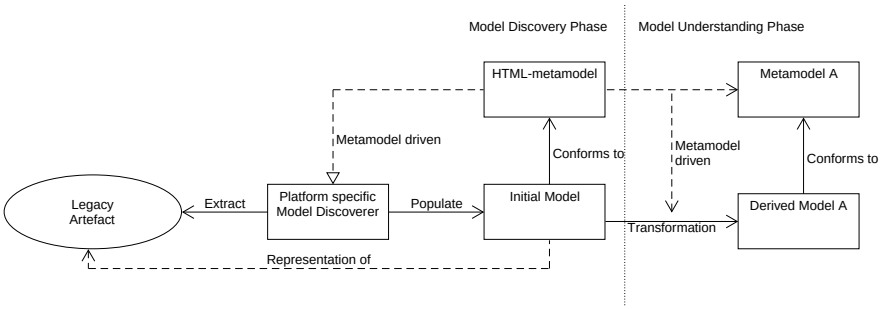

Fig. 4. The Model Discovery Phase and the Model Understanding Phase

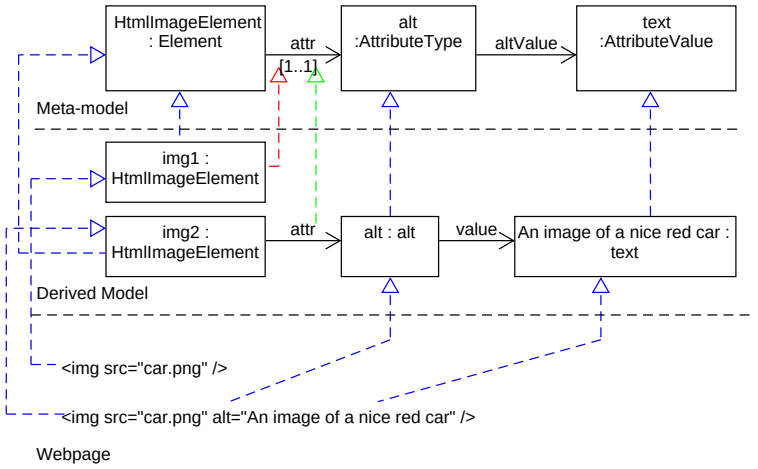

Fig. 5. The Model Validation Phase

HTML element is excluded from the process. This phase transforms the HTML element to a generic a set of nodes and edges representing the HTML element and its attributes with a graph-based visual syntax. Irrelevant information such as user events attached to the element are not included in the transformation, ensuring that the new representation of the HTML image element only contains the information that is necessary in order to confirm that it satisfies the *1.1.1 Non-text Content requirement*.

2.4 The Model Validation Phase

The final step in the Bottom-Up approach consists of checking if the derived model successfully conforms to the metamodel, if any part of the derived model is incorrectly typed, or doesn't satisfy the corresponding predicate, it is flagged as invalid. This encompasses the actual evaluation process that determines if the artefact contains any components that fail to meet the necessary requirements. The feedback presented to the user is purely graphical, consisting of a graph-based model representation of the HTML-document and a graph-based representation of the metamodel containing the constraints the HTML-document must satisfy. Any parts of the model representation that violates the metamodel, is highlighted red, warning the user of the violation. Concluding the example from the two previous steps, any HTML image element that is missing the *alt*-attribute will not conform to the metamodel reporting that the multiplicity predicate has not satisfied. Figure 5 illustrates the model validation phase using two image elements. The image element *img1* does not contain an alternative text, therefore it does not conform to the meta-model and is flagged as invalid.

The image element *img2* however, does contain an alternative text and therefore conforms to the meta-model.

3 Related Work

We now compare the Bottom-up approach to related approaches and existing web evaluation tools. The model driven reverse engineering framework *MoDisco* introduced in [5] is a tool intended to improve the design of legacy systems. It is only concerned with the first two phases of the MDRE process: the model discovery phase, and the model understanding phase. Upon obtaining the derived model, it allows it to be exported to external tools for software modernization, refactoring, retro documentation, quality analysis and so on. This means that MoDisco is adaptable to many different scenarios.

Software Modernization is a typical application of MDRE according to [4]. Software modernization involves migrating and redeploying legacy systems into modern software programs. This is achieved by extracting legacy platform specific components, transforming them to a generic derived model, then using the derived model to generate source code as functioning modern software components. In other words, the third phase of software modernization consists of a code generation process.

The Bottom-up approach suggests an alternative use of the derived model. Where [4] suggests to use the derived model as a starting point of a code generation process, and [5] suggests exporting the derived model to an external tool for further processing, the bottom-up approach preserves the derived model in the conceptual space, inserting it at the bottom level of a metamodelling hierarchy, thus providing the means to check if the model is a valid instance of the metamodel positioned at the level above. By confirming that the model is a valid instance of its metamodel, we can claim that the artefact it represents meets the necessary requirements described in the metamodel. A prototype of the bottom-up approach has been developed as a plug-in in WebDPF [10]. WebDPF is a multilevel metamodelling tool with model completion and simulation functionality. The plug-in extends WebDPF with the ability to create model discoverers and automatically validate web pages. A demonstration can be found at [2].

AChecker presented in [8] is a non-commercial web-based tool for evaluating the accessibility of a web site. It attempts to overcome the transparency problem by distributing the tool as an open source software, this is however considered an inadequate solution in [6], since it is only transparent to those with the necessary technical skills. AChecker does nonetheless provide an adequate amount of documentation of what is included in the evaluation process, and how it is evaluated, thus providing a potential solution to the Completeness and Correctness problem. *SortSite 5* is a commercial web site testing tool from PowerMapper [13], that validates a web page based on its level of accessibility as well as a number of other factors. It does not explain how it evaluates a web page or what it includes in the evaluation process. This means that it does not provide a solution to any of the challenges described in section 1. *Wave* is another commercial web

accessibility evaluation tool which is part of the WebAIM project. The WebAim project was started at the State University of Ohio in 1999 and focuses on improving web accessibility for individuals with disabilities. Wave provides limited information about what is included in the evaluation process, and no information about how it evaluates the web page. It does however present the results graphically, and provides according to [6], an adequate solution to the specificity problem. Since the evaluation process in the Bottom-Up approach resides solely in the observable conceptual space as a model conformance check, we can state that this approach is transparent. Furthermore, the metamodel included in the evaluation process, contains a sufficient description to determine how complete the evaluation process is, and how each element is evaluated. Table 1 gives an overview of the different tools, and which challenges they are able to overcome. Further details can be found in [6].

Challenge	**AChecker**	**SortSite 5**	**Wave**	**Bottom Up Approach**
Transparency Problem	No	No	No	**Yes**
Completeness Problem	**Yes**	No	No	**Yes**
Correctness Problem	**Yes**	No	No	**Yes**
Specificity Problem	Partly	Partly	**Yes**	**Yes**
Ambiguity Problem	N/A	N/A	N/A	**Yes**

Table 1. Overview of which challenges the existing web evaluation tools have overcome

4 Conclusion and Further Work

The Bottom-Up approach and the validation tool indicate that it may serve as a viable automated web evaluation tool that addresses several critical concerns. Based on what is presented in [6], and its ability to detect violations in a web page, it can be declared that the web evaluation tool has successfully fulfilled its intended purpose. Moreover, the approach proposed in this paper is to the best of the authors knowledge, introducing a new possible application for MDE. This solution has made it possible to formalize a set of requirements as a metamodel, then use that metamodel to evaluate artefacts belonging to the domain under study. In theory the solution can use a metamodel to evaluate any piece of accessible data. This could be considered as an innovative technique for reducing the gap between the conceptual world of modelling and the domain under study. Finally, by reducing the evaluation process to a conformance check between a model and it's metamodel, the process becomes transparent to the user.

We now envision some possible extensions of the bottom up approach.

Model Verification: Model conformance is just the tip of the iceberg when discussing model validation and verification, another factor that should be examined is the models consistency. This is addressed in the satisfiability problem which investigates if a meta-model contains any contradicting constraints, if so it will be impossible for a model to successfully conform to the meta-model. This problem could be applied to the metamodel describing the requirements, where

it could be determined if the artefact can actually satisfy all of the requirements included in the metamodel, or if changes to the requirements conflict with other existing ones. In the future we will use similar techniques as presented in [14] to check the satisfiability of the metamodel.

Model Adequacy: The Bottom-Up approach does not evaluate the actual artefact, but rather an abstract model representation of it. This means that in order for the evaluation process to present accurate results, the derived model needs to be adequate. An adequate derived model must contain all the necessary information that is relevant to the evaluation process. In order to ensure that the model is adequate, some sort of automated mechanism should be included in solution. The problem with model adequacy was first addressed in [11].

Completing the Metamodel: The metamodel described in this paper only covered 1 of the 35 mandatory requirements described in the WCAG 2.0 guidelines. However in [6] five out of the nine requirements included in the thesis were successfully included in the evaluation process and therefore supported by the tool. In order for the bottom-up approach to function as a complete web evaluation tool, the metamodel must include all of the mandatory requirements.

References

1. Atkinson, C., and Kühne, T.: The essence of multilevel metamodeling. International Conference on the Unified Modeling Language, pp19–33, Springer (2001)
2. Bottom up model validation approach: www.youtube.com/watch?v=qqUtLjSpj90
3. Brajnik, G.: Comparing accessibility evaluation tools: a method for tool effectiveness. Universal Access in the Information Society(2004)
4. Brambilla, M., Cabot, J., and Wimmer, M.: Model-driven software engineering in practice. Volume 1 Morgan & Claypool Publishers. (2012)
5. Bruneliere, H., Cabot, J., Dup, G., and Madiot, F.: Modisco: A model driven reverse engineering framework. Information and Software Technology, 56(8) (2014)
6. Calder, T.: A Model Driven Approach to Web Page Evaluation. Master Thesis, Bergen University College (2016), available at http://dpf.hib.no/publications/
7. Chikofsky, E. J., Cross, J. H., et al.: Reverse engineering and design recovery: A taxonomy. Software, IEEE, 7(1):13–17 (1990)
8. Gay, G. and Li, C. Q.: Achecker: open, interactive, customizable, web accessibility checking. In Proceedings of W4A, ACM DL (2010)
9. Fowler, M.: Domain-specific languages. Pearson Education (2010).
10. Rabbi F., Lamo Y., Yu I., Kristensen L.: WebDPF: A Web-based Metamodelling and Model Transformation Environment. Modelsward (2016).
11. Rugaber, S. and Stirewalt, K. Model-driven reverse engineering. Software, IEEE, 21(4):4553. (2004)
12. Sánchez-Cuadrado, J., De Lara, J., and Guerra, E.: Bottom-up meta-modelling: An interactive approach. Proceedings of Models 2012 in LNCS vol. 7590, pp. 3-19
13. PowerMapper: www.powermapper.com/products/sortsite/. Accessed: 2016-04-01
14. Wang, X., Rutle, A., Lamo, Y. Towards User-Friendly and Efficient Analysis with Alloy, In CEUR workshop proceedings, Vol. 1514, MoDevva (2015)
15. Wave, web accessibility evaluation tool: wave.webaim.org. Accessed: 2016-04-07
16. World Wide Web Consortium and others: Web content accessibility guidelines (WCAG) 2.0 World Wide Web Consortium (2008)