Disentangling Crosscutting in AOSD: A Conceptual Framework

Klaas van den Berg
Software Engineering Group, University of Twente
P.O. Box 217, 7500 AE, Enschede
The Netherlands
+31 (0)53 489 3783
k.g.vandenberg@ewi.utwente.nl

José María Conejero
Quercus SEG, University of Extremadura
Avda. Universidad s/n, C.P. 10071, Cáceres
Spain
+34 927 257268
chemacm@unex.es

ABSTRACT

Crosscutting is usually described in terms of scattering and tangling. However, the distinction between these three concepts is vague, sometimes leading to ambiguous statements and confusion. We propose a conceptual framework for crosscutting. Crosscutting is clearly distinguished from scattering and tangling. The definitions of these concepts are formalized and visualized with matrices and matrix operations. We discuss the relation between the concepts crosscutting, decomposition and coupling.

Categories and Subject Descriptors

D.1.5 [**Software Engineering**]: Object-oriented programming

D.2.1 [**Software Engineering**]: Requirements Engineering – *Methodologies.*

General Terms

Theory.

Keywords

Aspect Foundations, Scattering, Tangling, Crosscutting, Crosscutting Concerns

1. INTRODUCTION

One of the key principles in Aspect Oriented Software Development (AOSD) is Separation of Concerns (SOC). This principle is described in many publications [5][3]. Related with this principle is the problem of crosscutting concerns. Crosscutting is usually described in terms of scattering and tangling, e.g. crosscutting is the scattering and tangling of concerns arising due to poor support for their modularization. However, the distinction between these three concepts is vague, sometimes leading to ambiguous statements and confusion:

" .. the term "crosscutting concerns" is often misused in two ways: To talk about a single concern, and to talk about concerns rather than representations of concerns. Consider "synchronization is a crosscutting concern": we don't know that synchronization is crosscutting unless we know what it crosscuts. And there may be representations of the concerns involved that are not crosscutting. " (Kiczales, 2005 [6])

When talking about aspect orientation, we use concepts for which we have some intuition based on our specific experience. We share these concepts with others who may have a similar intuition usually based on other experience. However, the definitions of the concepts are sometimes vague, and sometimes not consistent with other concepts. Vague definitions imply that it is not always possible to decide when a certain concept applies. When do we have just scattering, when do we have just tangling, when do we have crosscutting and when not? Whenever possible, we should give definitions that are more precise.

The goal of this paper is to come up with general and consistent definitions and not to discuss specific examples - although some should fit somehow in this general framework. Especially, redefinition of existing concepts may lead to confusion and a lot of discussion. In this paper, we describe a conceptual framework with precise definitions of scattering, tangling and crosscutting. The description of crosscutting presented here is similar to some descriptions in the work of Masuhara & Kiczales (2003) [8] and of Mezini & Ostermann (2003) [10].

The paper is structured as follows. In section 2, we introduce the crosscutting pattern with definitions about crosscutting, tangling and scattering. In section 3, we describe how to represent and visualize crosscutting in a crosscutting matrix and how to derive this matrix from the dependency matrix using a scattering and tangling matrix. In section 4, we discuss the relationship between crosscutting and related concepts as coupling and decomposition. The section 5 consists of a particular example of using the crosscutting pattern with partitioning of the source. Finally, in section 6, we present the conclusions of the paper.

2. CROSSCUTTING PATTERN

In this section, we describe the assumption for the crosscutting pattern and the pattern itself. Furthermore, we focus on definitions of crosscutting, tangling and scattering. We describe matrix representations of these definitions.

2.1 Assumption

Our proposition is that tangling, scattering and crosscutting can only be defined in terms of 'one thing' with respect to 'another thing': at least two domains or two levels or two layers or two phases are related with each other in some way. For example:

- The two levels could refer to on one-hand concerns and on the other-hand *representations* of concerns, as stated in the citation in the introduction.
- The term *domain* could be used in mathematical sense where we have a mapping from one domain to another domain.
- The term *phase* could refer to phases in the software development cycle, such as concern modelling, requirements analysis, architectural design, detailed design and implementation.

- The term *level* could refer to levels in the Model Driven Architecture [9] (e.g. CIM, PIM and PSM).
- The term *layer* could refer to a multi-layer architecture with for example a User Interface Layer, Application Logic Layer, and Data Layer.

We use here the general terms *source* and *target* (as in [9]) to denote two consecutive domains, phases, levels or layers.

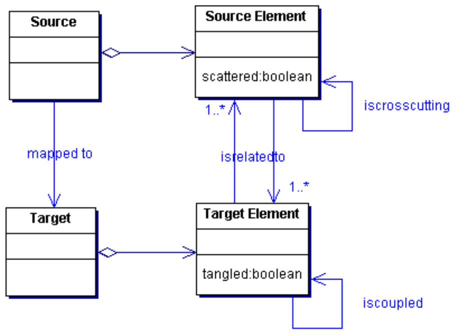

Figure 1 Concept Diagram of Crosscutting Pattern (without Mapping Concepts)

In the Crosscutting Pattern, elements in the source are related to elements in the target (see Figure 1). We use the term *pattern* as in design patterns [4], in the sense of being a general description of frequently encountered situations [8], [10]: e.g. we have phrases as "one thing *with respect to* another thing".

The relation between Source and Target in the pattern is *symmetric*. The roles may be interchanged. The definitions of tangling, scattering and crosscutting are relative to the place of source and target in the crosscutting pattern, denoted as (source x target).

Some examples of source and target elements in the crosscutting pattern are the following: concern x module, concern x requirement, concern x architectural element, requirement x module, design element x implementation element, concern x implementation element.

There is a mapping between source elements and target elements. The terms crosscutting, tangling and scattering are defined as special cases of these mappings. This is explained in the following section.

2.2 Concepts based on Crosscutting Pattern

We can extend the Crosscutting Pattern of the previous section with related concepts. The concept diagram is given in Figure 2.

SourceToTargetMapping is the relation between source elements and target elements. The relation can have different types. The mapping has a multiplicity. It could be 1:1 or 1:many. In case of 1:many mappings we have scattering, defined as follows: *Scattering occurs when, in a mapping between source and target, a source element is related to multiple target elements.*

TargetToSourceMapping is the relation between target elements and source elements, as a result of a mapping of source to target. This relation (here also called mapping) is the reverse of the mapping above. The multiplicity could be 1:1 or 1:many. In case of 1:many mappings we have tangling, defined as follows: *Tangling occurs when, in a mapping between source and target, a target element is related to multiple source elements.* The nature of tangling can be different and there may be different levels of tangling intensity. We will not discuss this issue here, but only consider the multiplicity. We say that: *Two source elements are tangled if these elements are mapped onto the same target element.*

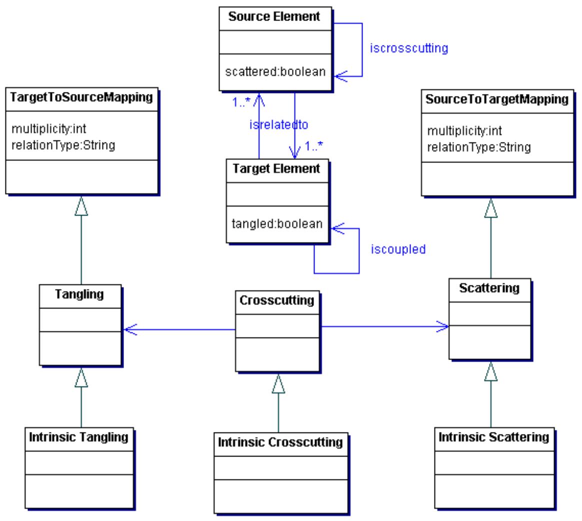

Figure 2 Concept Diagram of Crosscutting Pattern (with Mapping Concepts)

There is a specific combination of scattering and tangling which we call crosscutting, defined as follows: *Crosscutting occurs when, in a mapping between source and target, a source element is scattered over target elements and where in at least one of these target elements, some other source elements are tangled.*

Now, we can also give a definition for the crosscutting of two source elements: *Source element s1 crosscuts source element s2 if s1 is scattered over target elements, and in at least one of these target elements, s1 is tangled with source element s2.*

We do not require that the second source element is scattered. In that sense, our definition is not symmetric and less restrictive than Masuhara & Kiczales's definition [7].

The concepts intrinsic scattering, tangling and crosscutting are explained in section 4.

2.3 Case Analysis of Crosscutting

In the previous section, we defined scattering, tangling and crosscutting. Now we discuss a case analysis of possible combinations. Assuming that the properties tangling, scattering, and crosscutting may be true or false, there are 8 combinations (see Table 1). However, crosscutting requires tangling and scattering, which eliminates 3 of these combinations (not feasible). There are five feasible cases listed in the table. Each case addresses a certain mapping from source to target.

In case 4, we have scattering and tangling in which no common elements are involved. We show an example of this situation in Table 2 (Section 3.1). With our definition of crosscutting, we discriminate between the cases with just tangling, just scattering and on the other hand crosscutting. *Our proposition is that tangling and scattering are necessary but not sufficient conditions* *for crosscutting*. The rationale for disentangling these concepts is that there may be different solutions for each of these situations.

Table 1. Combinations of the Properties Tangling, Scattering and Crosscutting

property	tangling	scattering	crosscutting	Feasibility
Case 1.	no	no	no	Feasible
Case 2.	yes	no	no	Feasible
Case 3.	no	yes	no	Feasible
Case 4.	yes	yes	no	Feasible
Case 5.	yes	yes	yes	Feasible
Case 6.	no	no	yes	Not feasible
Case 7.	no	yes	yes	Not feasible
Case 8.	yes	no	yes	Not feasible

3. MATRIX REPRESENTATION

In this section, we describe how crosscutting can be represented in matrices. We start with the dependency matrix showing the mapping between source and target. From this matrix, we derive the crosscutting matrix, illustrated with some examples. Then, we describe how the crosscutting matrix can be constructed from the dependency matrix with some auxiliary matrices.

3.1 Definitions of matrices

The relation between source elements and target elements can be represented in a dependency matrix. In some sense, the dependency matrix displays the *traceability* between source and target elements. *A dependency matrix (source x target) represents the dependency relation between source elements and target elements (inter-level relationship)*. In the rows, we have the source elements, and in the columns, we have the target elements. In this matrix, a cell with 1 denotes that the source element (in the row) is mapped to the target element (in the column). Scattering and tangling can easily be visualized in this matrix (see the examples below).

We define a new auxiliary concept *crosscutpoint* used in the context of dependency diagrams, to denote *a matrix cell involved in both tangling and scattering*. If there are one or more crosscutpoints then we say we have crosscutting.

Crosscutting between source elements for a given mapping to target elements, as shown in a dependency diagram, can be represented in a crosscutting matrix. *A crosscutting matrix (source x source) represents the crosscutting relation between source elements, for a given source to target mapping (represented in a dependency matrix).* In the crosscutting matrix, a cell with 1 denotes that the source element in the row is crosscutting the source element in the column. In section 3.2 we explain how this crosscutting matrix can be derived from the dependency matrix.

A similar view on crosscutting can be found in [8] (see Figure 3). Crosscutting is defined as follows: "For a pair of modules m_A and m_B we say that *m_A crosscuts m_B with respect to X* if and only if their projections onto X intersect, and neither of the projections is a subset of the other."

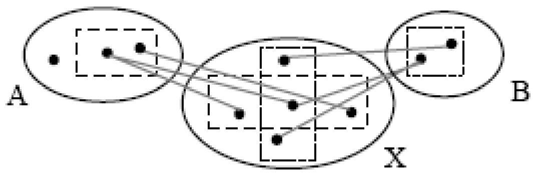

Figure 3. Crosscutting of modules [8]

In our terminology, there is a source consisting of A and B and a target X. The *intersection* in this definition is similar to our definition of crosscutpoint (*involved both in*). Our definition is less restrictive because it does not require the subset relation: we only require that the cardinality of the projection of m_A onto X is larger than 1. This also implies that in our definition crosscutting is not a symmetric property.

A crosscutting matrix should not be confused with a coupling matrix. A *coupling matrix* shows coupling relations between elements at the same level (intra-level dependencies). In some sense, the coupling matrix is related to the design structure matrix [7]. A crosscutting matrix shows crosscutting relations between elements at one level with respect to a mapping onto elements at some other level (inter-level dependencies).

We now give examples of the cases 4 and 5 presented in Table 1. In these examples, we give definitions of concepts used to describe the cases. We use the dependency matrix and the crosscutting matrix (from the previous section) to visualize the definitions and examples (S denotes a scattered source element - a grey row; NS denotes a non-scattered source element; T denotes a tangled target element - a grey column; NT denotes a non-tangled target element).

For case 4 (tangling, scattering, no crosscutting), an example mapping is shown in Table 2. In this example, we have one scattered source element s[1] and one tangled target element t[3]. However, t[3] is not involved in the tangling and scattering. We apply our definition of crosscutting and arrive to the crosscutting matrix. In this case, there is no crosscutting.

Table 2. Example dependency and crosscutting matrix with tangling, scattering and no crosscutting

dependency matrix

		target				
		t[1]	t[2]	t[3]	t[4]	
source	s[1]	1	1	0	1	S
	s[2]	0	0	1	0	NS
	s[3]	0	0	1	0	NS
		NT	NT	T	NT	

crosscutting matrix

		source		
		s[1]	s[2]	s[3]
source	s[1]	0	0	0
	s[2]	0	0	0
	s[3]	0	0	0

For case 5 (tangling, scattering, crosscutting), an example mapping is shown in Table 3. Is this example, we have one scattered source element s[1] and one tangled target element t[3].

Moreover there is one crosscutpoint at matrix cell [1,3] (dark grey cell). Again applying our definition, we arrive to the crosscutting matrix. Source element s[1] is crosscutting s[3] (because s[1] is scattered over [t[1], t[3], t[4]] and s[3] is in the tangled one of these elements, namely t[3]. The reverse is not true: the crosscutting relation is not symmetric. This illustrates our proposition about crosscutting not being symmetric. The example is depicted in the following diagrams (Table 3).

Table 3. Example dependency and crosscutting matrix with tangling, scattering and one crosscutting

dependency matrix

		Target				
		t[1]	t[2]	t[3]	t[4]	
source	s[1]	1	0	1	1	S
	s[2]	0	1	0	0	NS
	s[3]	0	0	1	0	NS
		NT	NT	T	NT	

crosscutting matrix

		source		
		s[1]	s[2]	s[3]
Source	s[1]	0	0	1
	s[2]	0	0	0
	s[3]	0	0	0

3.2 Constructing crosscutting matrices

In this section, we describe how to derive the crosscutting matrix from the dependency matrix. We use a more general example than the previous cases. We now show an example with more than one crosscutpoints, in this example 8 points (see Table 4; the dark grey cells). There are now 10 crosscutting relations between the source elements.

Table 4. Example dependency matrix with tangling, scattering and several crosscuttings

Dependency matrix

		target						
		t[1]	t[2]	t[3]	t[4]	t[5]	t[6]	
source	s[1]	1	0	0	1	0	0	S
	s[2]	1	0	1	0	1	1	S
	s[3]	1	0	0	0	0	0	NS
	s[4]	0	1	1	0	0	0	S
	s[5]	0	0	0	1	1	0	S
		T	NT	T	T	T	NT	

Crosscutting matrix

		source				
		s[1]	s[2]	s[3]	s[4]	s[5]
source	s[1]	0	1	1	0	1
	s[2]	1	0	1	1	1
	s[3]	0	0	0	0	0
	s[4]	0	1	0	0	0
	s[5]	1	1	0	0	0

The crosscutting matrix shows that the crosscutting relation is not symmetric. For example, s[1] is crosscutting s[3], but s[3] is not crosscutting s[1] because s[3] is not scattered (scattering is a necessary condition for crosscutting).

Based on the dependency matrix, we define some auxiliary matrices: the *scattering matrix* (source x target) with just scattering, and the *tangling matrix* (target x source) with just tangling. These two matrices are defined as follows:

- In the scattering matrix a row contains only dependency relations from source to target elements if the source element in this row is scattered (mapped onto multiple target elements); otherwise the row contains just zero's (no scattering relation).

- In the tangling matrix a row contains only dependency relations from target to source elements if the target element in this row is tangled (mapped onto multiple source elements); otherwise the row contains just zero's (no tangling relation).

For our example in Table 4, these matrices are the following (see Table 5).

Table 5. Example scattering and tangling matrices for dependency matrix in Table 4

Scattering matrix

		target					
		t[1]	t[2]	t[3]	t[4]	t[5]	t[6]
source	s[1]	1	0	0	1	0	0
	s[2]	1	0	1	0	1	1
	s[3]	0	0	0	0	0	0
	s[4]	0	1	1	0	0	0
	s[5]	0	0	0	1	1	0

Tangling matrix

		source				
		s[1]	s[2]	s[3]	s[4]	s[5]
target	t[1]	1	1	1	0	0
	t[2]	0	0	0	0	0
	t[3]	0	1	0	1	0
	t[4]	1	0	0	0	1
	t[5]	0	1	0	0	1
	t[6]	0	0	0	0	0

We now define the crosscutting product matrix, showing the frequency of crosscutting relations. *A crosscutting product matrix (source x source) represents the frequency of crosscutting relations between source elements, for a given source to target mapping.* The crosscutting product matrix is not necessarily symmetric. The *crosscutting product matrix* ccpm can be obtained through the matrix multiplication of the scattering matrix sm and the tangling matrix tm: $ccpm = sm \, . \, tm$ where $ccpm_{ik} = \; sm_{ij} \, tm_{jk}$

In this crosscutting product matrix, the cells denote the frequency of crosscuttings. This can easily be used for quantification of crosscutting (crosscutting metrics).

The frequency of crosscuttings in this matrix should be seen as an upper bound. In actual situations, the frequency can be less than the frequency from this matrix analysis, because in the matrix we abstract from scattering and tangling specifics.

In the crosscutting matrix, a matrix cell denotes the occurrence of one or more crosscuttings; it abstracts from the frequency of crosscutting.

The *crosscutting matrix* ccm can be derived from the crosscutting product matrix ccpm using a simple conversion: ccm_{ik} = if $(ccpm_{ik} > 0) \wedge (i \neq j)$ then 1 else 0

These two crosscutting matrices for the example are given in Table 6. In this example, there are no cells in the crosscutting product matrix larger than 1, except on the diagonal where it denotes a crosscutting relation with itself, which we disregard here. In the crosscutting matrix, we put the diagonal cells to 0.

Table 6. Example crosscutting product matrix and corresponding crosscuttings matrix

Crosscutting product matrix

		source				
		s[1]	s[2]	s[3]	s[4]	s[5]
source	s[1]	2	1	1	0	1
	s[2]	1	3	1	1	1
	s[3]	0	0	0	0	0
	s[4]	0	1	0	1	0
	s[5]	1	1	0	0	2

Crosscutting matrix

		source				
		s[1]	s[2]	s[3]	s[4]	s[5]
source	s[1]	0	1	1	0	1
	s[2]	1	0	1	1	1
	s[3]	0	0	0	0	0
	s[4]	0	1	0	0	0
	s[5]	1	1	0	0	0

We formalized the definitions in a functional programming language (executable mathematics). For convenience, these formulas can be put in an Excel sheet using the function for matrix multiplication. By filling in the cells of the dependency matrix, the other matrices are calculated automatically.

4. CROSSCUTTING AND RELATED CONCEPTS

In this section, we discuss how the crosscutting is related with the concepts of decomposition and coupling. The definitions of crosscutting are based on a mapping from source to target (represented in the dependency matrix with source and target elements). In some cases it is possible to avoid tangling, scattering and crosscutting by choosing another decomposition of source and target. The possibilities are determined by the expressive power of the languages in which the source and target are expressed.

"Crosscutting models are themselves not the problem. The problem is that our languages and decomposition techniques do not properly support crosscutting modularity." (Mezini & Ostermann, 2003 [10])

The role of the source and target languages can be made clear in an extension to the crosscutting pattern (see Figure 4). (c.f. metamodel transformation pattern [9]). A source can be described using several languages at the same time. This also applies to the target. In case where limitations in the expressive power of the languages are the cause of tangling, scattering and/or crosscutting we use the terms *intrinsic tangling*, *intrinsic scattering* and *intrinsic crosscutting.*

Here we just present the concepts: in specific cases there must be debate and arguments to decide whether or not there are essential limitations in the languages. The extension of a language with new constructs and new composition operators - such as aspects or composition filters - may change the (de)composition of source and target. Hence, it will affect the dependency matrix and the related analysis of scattering, tangling and crosscutting.

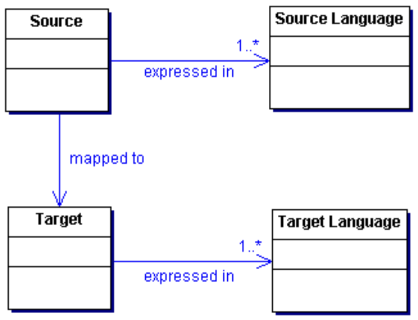

Figure 4. Concept diagram with the role of languages in the Crosscutting Pattern

In this paper, we assume the existence of dependency matrices as our starting point. The creation of the actual decomposition and modularization is a very important research issue that is not addressed here. There are problems: the problem of decomposition (e.g. modularisation) of source and target, dominant decompositions, composition operators, granularity of decomposition, the type of dependency relations between source elements and target elements, but also the intra-level dependency relations of source elements and of target elements.

Moreover, usually there are many alternative decompositions both in source and target, and many alternative mappings between source and target. One has to compare combination of alternative compositions on quality attributes such as adaptability, reusability, maintainability. It is clear that the quality of the dependency matrix itself determines the quality of the analysis of crosscutting based on this matrix.

Elements at a certain level can be decomposed in more basic elements at the same level (e.g. in the composite pattern). This may affect the set up of the dependency matrix: one has to choose at what degree of granularity the relation between source and target will be analysed. Composite elements occur at any level, for example in implementation components but also in concern modelling and in separation of concerns.

Elements at a certain level usually have some relationship with other elements at the same level (intra-level relationships): they are coupled. There are many coupling types: generalisation/specialisation, aggregation, data coupling, control coupling, message coupling, and so on. In case of a dependency relation of a source element and a target element, which itself is coupled to a second target element, one could conceive also a dependency relation between the source element and the second target element.

However, this is another dependency relation, which we call an *indirect dependency* based on a pseudo-transitivity.

Assume source element s[i] has a dependency relation R with target element t[k].

Moreover, target element t[k] is coupled with target element t[m] represented with dependency relation R'. Then the indirect dependency relation is

$$(s[i]\ R\ t[k]) \wedge (t[k]\ R'\ t[m]) \Rightarrow (s[i]\ R\ t[m])$$

One should clearly distinguish the normal inter-level dependency relation from this indirect dependency relation.

5. EXAMPLE: PARTITIONING OF SOURCE AND/OR TARGET

In this section, we give an example how crosscutting and the dependency matrix can be used to clarify statements about crosscutting. We consider the case where elements of source and/or target are grouped in certain categories based on the type of the elements. In requirements analysis, one could distinguish functional requirements and non-functional requirements. We illustrate this in an example (c.f. [11]).

We consider the case in which we have a mapping between requirements (our source) onto architectural elements (our target). The source has two parts: the functional requirements (FR) and the non-functional requirements (NFR). In this example, the crosscutting matrix has four quadrants. These quadrants represent the following cases:

- Functional Requirements crosscut Functional Requirements
- Functional Requirements crosscut Non-Functional Requirements
- Non-Functional Requirements crosscut Functional Requirements
- Non-Functional Requirements crosscut Non-Functional Requirements

Table 7. Example dependency matrix with partitioning of source

dependency matrix

		target					
		t[1]	t[2]	t[3]	t[4]	t[5]	t[6]
NFR	s[1]	1	0	0	1	0	0
	s[2]	1	0	1	0	1	1
FR	s[3]	1	0	0	0	0	0
	s[4]	0	1	1	0	0	0
	s[5]	0	0	0	1	1	0

crosscutting matrix

		NFR		FR		
		s[1]	s[2]	s[3]	s[4]	s[5]
NFR	s[1]	0	1	1	0	1
	s[2]	1	0	1	1	1
FR	s[3]	0	0	0	0	0
	s[4]	0	1	0	0	0
	s[5]	1	1	0	0	0

As an example, we use the dependency matrix from the previous section (see Table 7). We assume that s[1] and s[2] are non-functional requirements, and s[3], s[4] and s[5] are functional requirements. In this example, we have no functional requirements crosscutting other functional requirements. Crosscuttings are for example:

- NFR s[1] is crosscutting NFR s[2] and
- NFR s[1] is crosscutting FR s[3] and FR s[5]

Again, it is important to note that the crosscuttings of requirements are relative to the mapping onto - in this case - architectural elements. Moreover, the actual decompositions of requirements and architectural elements are crucial in the analysis of crosscutting based on the dependency matrix.

6. CONCLUSION

In this paper, we proposed a conceptual framework for describing crosscutting. We introduced a crosscutting pattern with a mapping from a source to a target. With source and target, we abstract from specific levels or phases in software development. We defined crosscutting, tangling and scattering as separated cases based on different mappings between source and target. We introduced the dependency matrix and crosscutting matrix to visualize the definitions. We showed that it is possible to formalize these definitions. The proposed definitions are similar to definitions of crosscutting in some other publications, e.g. [8], although our definition is not symmetric and less restrictive. The crosscutting pattern can be applied in different specific cases such as a partitioning of the source Moreover, it provides the ability to define crosscutting metrics, e.g. to quantify the intensity of crosscuttings

A very interesting application is the cascading of crosscutting patterns, which can be used to model crosscutting relations across several levels, for example from concern modelling, to requirements, architectural design to detailed design and implementation. As such, it provides an approach for traceability analysis.

Some concepts in this framework are new, reused or redefined from other approaches and perspectives. We should apply this framework in different concrete situations in order to establish the suitability of the chosen concepts and definitions. The following topics could be investigated in terms of the crosscutting pattern: Aspectual requirements analysis [11], Aspectual architectural design [13], Concern modelling [12], and Concern manipulation environment [2]. Experience with the application of the framework could result either in an adjustment of the concepts in the framework or to another view on problems in the application area. The conceptual framework presented here focuses on crosscutting, tangling and scattering. The framework should be extended with other AOSD concepts.

ACKNOWLEDGEMENT

This work has been carried out in conjunction with the AOSD-Europe Project IST-2-004349-NoE [1] and also partially supported by CICYT under contract TIC2002-04309-C02-01. The authors would like to thank Gurcan Gulesir and other members of the Software Engineering Group at the University of Twente for the discussion of this framework.

REFERENCES

[1] AOSD-Europe (2004). IST Project Proposal 004349, Annex I - Description of Work, 1 September 2004.

[2] Concern Manipulation Environment (CME). See http://www.research.ibm.com/cme/

[3] Filman, R., et al., *Aspect-Oriented Software Development*. 2004: Addison-Wesley.

[4] Gamma, E., Helm, R., Johnson, R., & Vlissides, J. (1995). Design patterns. Elements of reusable object-oriented software: Addison-Wesley.

[5] Hürsch, W. and Lopes, C. (1995). Separation of Concerns. Technical Report, College of Computer Science, Northeastern University.

[6] Kiczales, G. *Crosscutting*. AOSD.NET Glossary 2005 [cited; Available from: http://aosd.net/wiki/index.php?title=Crosscutting

[7] Lopes, C.V. and S.K. Bajracharya. An analysis of modularity in aspect oriented design. In 4th international conference on Aspect-Oriented Software Development. 2005. Chicago, Illinois

[8] Masuhara, H. and G. Kiczales. Modeling Crosscutting in Aspect-Oriented Mechanisms. In ECOOP 2003. 2003. Darmstadt

[9] MDA (2003). MDA Guide Version 1.0.1, document number omg/2003-06-01.

[10] Mezini, M. and K. Ostermann. Modules for Crosscutting Models. In 8th International Conference on Reliable Software Technologies. 2003. Toulouse, France: LNCS 2655

[11] Rashid, A., A. Moreira, and J. Araujo. Modularisation and Composition of Aspectual Requirements. In Second AOSD Conference. 2003. Boston

[12] Sutton, S. and I. Rouvellou, Concern Modeling for Aspect-Oriented Software Development, in Aspect-Oriented Software Development, R. Filman, et al., Editors. 2004, Addison Wesley

[13] Tekinerdogan, B. ASAAM: Aspectual Software Architecture Analysis Method. in WICSA 4th Working IEEE/IFIP Conference on Software Architecture. 2004: IEEE