ABSTRACT
Aspect-Oriented Programming (AOP) has emerged in recent years as a new paradigm for software development. PRISMA is an approach for developing complex and large software systems. It combines the Aspect-Oriented Software Development (AOSD) and the Component-Based Software Development (CBSD) in an elegant and novel way achieving a better management of crosscutting-concerns and software reusability. PRISMA approach proposes the separation of concerns from the very beginning of the software life-cycle in order to introduce them as reusable aspects of software architecture. In this paper, we focus on how the PRISMA modelling tool supports the graphical modelling of aspects and weavings.

Categories and Subject Descriptors
D.2.11 [Software Engineering]: Software Architectures, Domain-specific architectures, Information hiding, Languages.

General Terms
Design, Languages, Theory.

Keywords
Keywords are your own designated keywords.

1. INTRODUCTION
PRISMA approach has a formal model that uses AOSD principles to describe software architectures. There are three kinds of architectural elements: components, connectors and systems which can be analyzed from two different views: internal (white box view) and external (black box view) (see Figure 1). The white box view shows an architectural element as formed by a set of aspects and their synchronization relationships (weavings). The black box view encapsulates the architectural element functionality and publishes a set of services that it offers to other architectural elements. A PRISMA aspect represents a specific concern for the software architecture under development. Thus, from the very beginning of the specification all the behaviour is modularized in aspects, so that they can be imported by one or more architectural elements in order to properly manage crosscutting-concerns. The kind of aspect (safety, coordination, distribution, etc.) that forms an architectural element depends on the concerns of the software system that we are specifying. A full description of PRISMA model can be find in [2].

Figure 1. Internal and external view of PRISMA

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC ’06, April, 23-27, 2006, Dijon, France.
Copyright 2006 ACM 1-59593-108-2/06/0004...$5.00.
the weaving information is usually specified in two locations: in the aspect by means of both pointcuts and the methods of the advice (after, before, instead); and, in the kernel class by the joinpoints. The reuse of aspects is lost due to the fact that aspect specifications are strongly dependent of a determined class.

Safety Aspect Smotion

Attributes
Constant
  minimum, maximum: integer, NOT NULL;
Services
  begin()
  in check(Direction, integer, output Secure: boolean);
  ...
Operations
  transaction
  DANGEROUSCHECKING(input Degrees: integer, output
  CurrentSpeed: integer, output Secure: boolean);
  ...
Protocol
  CHECKING = begin.CHECKING;
  CHECKING = check (Degrees, Secure);
  DANGEROUSCHECKING(Degrees, Secure) + end;
End_Safety Aspect Smotion;

Figure 2. Example of PRISMA Safety aspect

In PRISMA, the weaving indicates that the execution of an aspect service can generate the execution of services in other aspects. In order to preserve the independence and the reuse of an aspect from the other aspects, the weaving specification is defined outside the aspect and inside the architectural element (see Figure 2). The weaving methods are operations that describe the causality of the weaving services. The weaving methods that are typical of the AOP are the following: after, before and around (or instead). In addition, PRISMA extends the weaving operators with their respective conditionals: afterif (condition), beforeif (condition), instead (condition).

Since a PRISMA architectural element is formed by a set of aspects, the weaving specification is part of the architectural element specification. We achieve not only the reusability of the aspects in different architectural elements but also the specification of different behaviours of an architectural element by importing the same aspects and defining different workings. This is possible thanks to the fact that the weaving is external to the aspects.

1.2 Modelling PRISMA Aspects

A lightweight extension of Visio that we have called EGV-PRISMA has been developed by means of a template to support the needs of the PRISMA AOADL. Therefore, whenever a new system has to be modelled, the EGV-PRISMA template is instantiated to create a new document.

Visual modelling in Visio is accomplished by means of Stencils. In the development of the EGV-PRISMA, two stencils were specifically developed: Aspect and interface PRISMA (see Figure 3) and Main Structure PRISMA. The former has been defined to allow the modelling of those aspects that are currently supported by PRISMA and the weaving relationships. The latter shows the stencil for modelling components, connectors and systems along with the relationships made available to them, i.e., binding and attachment, to be properly glued together. In addition, aspects and weaving details are specified by means of forms (Figure 4).

2. Conclusions

We have sketched a graphical modelling support for PRISMA AOADL. EGV-PRISMA is a tool that allows us to visually model PRISMA elements. Aspects appear as internal view of components and connectors. Weavings are represented as relationships between aspects. Details of aspects and weaving are specified by means of provided forms. This tool constitutes part of the PRISMA framework oriented to automatic generation of code [3].

ACKNOWLEDGMENTS

This work has been funded by CICYT project (Spain), DYNAMICA TIC2003-07776-C02-02

3. REFERENCES