

Weaving behavioural models

Jacques Klein, Jean-Marc Jézéquel, Noël Plouzeau

► **To cite this version:**

Jacques Klein, Jean-Marc Jézéquel, Noël Plouzeau. Weaving behavioural models. In First Workshop on Models and Aspects, Handling Crosscutting Concerns in MDS at ECOOP 05, Jul 2005, Glasgow, United Kingdom. 2005. <hal-00795068>

HAL Id: hal-00795068

<https://hal.inria.fr/hal-00795068>

Submitted on 27 Feb 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Weaving Behavioural Models

Jacques Klein, Jean-Marc Jézéquel, Noël Plouzeau
 IRISA, Campus de Beaulieu, 35042 Rennes Cedex, France
 {jklein, jezequel, nplouzeau}@irisa.fr

1. Introduction

Languages for aspect-oriented programming (AOP) [5], such as AspectJ, are now popular, and the concepts used by the AOP community such as join points, pointcuts and advices are well-known. At the same time, in recent years, the aspect oriented software development (AOSD) approach has been developing itself beyond the programming activity. More particularly, the *Early Aspects Initiative* [9] advocates the management of crosscutting properties, i.e. aspects, at the early development stages of requirements engineering and architecture design to identify the impact of aspects as soon as possible. Some composition operators of aspects exist for these development stages [1][2], but they do not closely match standard AOP concepts (pointcuts, advices...). In this paper, we propose an automatic way for weaving behavioural aspects given as scenarios.

2. Scenarios and Behavioural Aspects

Scenario languages have been the subject of significant interest during the last decade. They are mainly used to describe behaviours of distributed systems at an abstract level or to capture requirements in early development stages with a clear, graphical, and intuitive representation. More precisely, we use scenarios expressed as Message Sequence charts (MSC) [4], which propose two levels of specifications. At the lowest level, basic MSCs (bMSCs) describe simple communication patterns between entities of the system. However, bMSCs alone do not have a sufficient expressive power: they can only define finite behaviours, without real alternatives. For this reason, MSCs have been extended with High-level MSCs, a higher level of specification. HMSCs allow the composition of bMSCs with composition operators such as sequence, alternative and loop as illustrated in figure 1. Roughly speaking, HMSCs are a kind of bMSC automata. Figure 1 shows a classic example of an account creation followed by a log in. More precisely: a user of the system must first enrol with the system. Once enrolled, he must log on to the system for each session. The alternative behaviours are also depicted: an enrolment can be rejected because the data are wrong. The log in can be rejected because the password is wrong. Finally, the id can be wrong, and then the user can enrol or try to log in again.

Let us note that, UML 2.0 Sequence Diagrams [8] are largely inspired by MSCs. (A comparison between UML2.0 and MSC-2000 is given in [3]).

A whole and consistent scenario of a system can be difficult to specify because several behaviours are the result of crosscutting requirements which are relatively independent of the scenario of the system (called nominal scenario). One of the principles of AOSD is to have a separation between the aspects and the nominal system. So, we will simply represent behavioural aspects by a set of independent scenarios, the problem being now to know where and how weave the aspects into the nominal scenario.

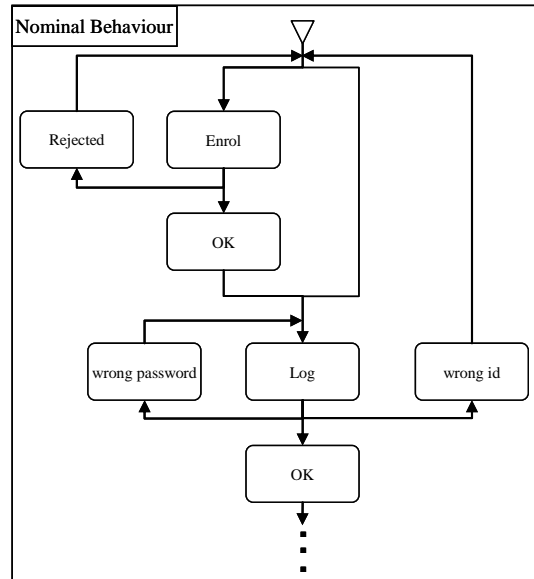


Figure 1 : An nominal scenario.

3. The Weaving Process

As in AspectJ but at the modelling level, we use the concepts of pointcuts and advices in our weaving process. Indeed, our aspects are composed of two parts. One part is the pointcut which matches specified patterns in the nominal scenario where the second part of the aspect, the advice, is woven. More precisely, the advice specifies the behaviour

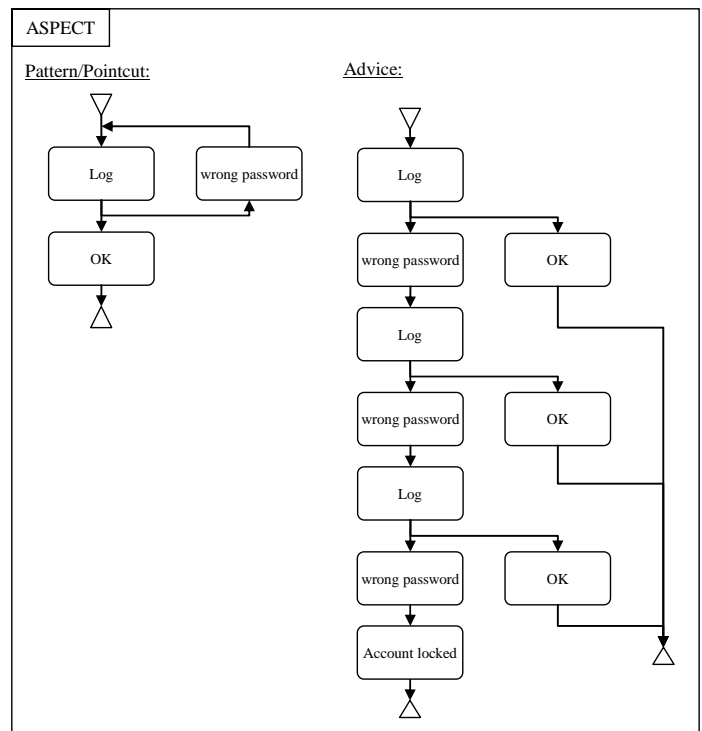


Figure 2: An aspect of security

