



AMPLE
Aspect-Oriented, Model-Driven, Product Line
Engineering
Specific Target Research Project: IST-33710

**Implementation platform and
documentation for using it**

ABSTRACT

This document gives an overview of the deliverable D3.3. The deliverable provides software packages that are necessary for deploying the implementation platform developed in workpackage 3 (Implementation) and its documentation. Installation instructions of the tools are given in this document.

Document ID:	AMPLE D3.3
Deliverable/Milestone No:	D3.3
Workpackage No:	WP3
Type:	Deliverable
Dissemination:	PU
Status:	final
Version:	V1.0
Date:	2009-10-05
Author(s):	Vaidas Gasiūnas, Mira Mezini (TUD), Jacques Noyé, Angel Núñez, Nicolas Anquetil (EMN), Ludger Fiege, Iris Groher, Christine Schwanninger, Markus Völter (Siemens), Pablo Sánchez, Lidia Fuentes (UMA)

Project Start Date: 01 October 2006, Duration: 3 years

1 Overview of the Implementation Platform

The implementation platform developed in the workpackage 3 consists of an aspect-oriented programming language ECAESARJ and a model-driven tool-suite openArchitectureWare.

ECAESARJ is a further development of the CAESARJ language [AGMO06], based on the ideas formulated in the deliverable D3.2 [GSN⁺07]. The programming language provides a lot of flexibility for decomposing software into modules. Like CAESARJ, the new language also implements virtual classes, propagating mixin composition, and path-dependent types, which enable type-safe flexible decomposition of classes and their methods. In addition to that, the language introduces distinction between required and provided interfaces of classes, implementing feature modules, and so enables more precise modular checking of feature implementations.

The pointcuts of CAESARJ have been replaced by a more general notion of events. The events integrate joinpoints, considered as implicit events, with explicitly triggered events. They are supplied with mixin composition semantics, which enables decoupling of event uses from event definition, extension and composition of an event. Events are defined as attributes of objects, which can be accessed polymorphically through object interfaces including abstract events. The events simplify description of inversion of control between feature modules, and serve as new extensible abstractions, whose definitions can be distributed among multiple features.

ECAESARJ also introduces explicit support for state machines and implements mixin composition semantics for them, thereby enabling decomposition of state machines by features. The state machine constructs also enable explicit organization of behaviour by states, and so support stable extension and composition of state-dependent behaviour.

The implementation platform includes the ECAESARJ compiler and a plugin, which integrates the compiler to the Eclipse and provides various IDE facilities. The plugin provides facilities to create, edit and execute ECAESARJ programs. It also adapts various JDT features to the ECAESARJ language, in particular various JDT tools for browsing and searching the source code. The functionality of the plugin is described in the ECAESARJ manual.

The openArchitectureWare is a quite popular tool suite for model-driven development. In the context of the project we extended the tool suite with aspect-oriented extensions. In particular, we developed the XWeave tool for aspect-oriented composition of models. Also, the model-to-model and model-to-code transformation languages of the toolsuite, xTend and xPand, were extended with support for aspects. The aspect-oriented extensions enable better modularization of variation at the level of models and transformations. The extensions developed in the project have been included to the standard openArchitectureWare distribution since its version 4.2.

The features of ECAESARJ and openArchitectureWare are described in the manuals, which are supplied as a part of this deliverable. Motivation of the tools and methodology of using them for software product line development are

described in the deliverable D3.4 [GMS⁺09]. As explained in D3.4, the tools are a part of a larger integrated tool suite of AMPLE project, supporting the proposed SPL development methodology.

2 Contents of the Deliverable

The deliverable contains the following items:

- Installation package of ECAESARJ plugin for Eclipse with an integrated ECAESARJ compiler. See file `ecaesarj-win-20090925.zip`.
- Installation package of openArchitectureWare 4.3.1 SDK Classic, including aspect-oriented extensions developed in AMPLE. See file `org.openarchitectureware.sdk.feature-4.3.1.20090107-2000PRD.zip`.
- User manual of ECAESARJ describing specific language features and plugin functionality. See file `ECAesarJ-Manual.pdf`.
- User manual of openArchitectureWare including descriptions of the extensions. See file `openArchitecture-4.3.1-Manual.pdf`.

3 Installation of Tools

Platform Requirements

- JDK 1.6 or later;
- Eclipse 3.4;
- The provided ECAESARJ compiler binaries depend on Windows XP or later. The source code of the compiler can also be compiled for any Unix-based operating system.

Installation of ECAesarJ To install ECAESARJ plugin, unpack the contents of `ecaesarj-win-20090925.zip` into the `dropins` folder of your Eclipse instance.

Installation of openArchitectureWare openArchitectureWare can be installed from an Eclipse update site located at:

<http://www.openarchitectureware.org/updatesite/milestone/4.3.1/classic/>

Alternatively openArchitectureWare can be installed by unpacking the contents of `org.openarchitectureware.sdk.feature-4.3.1.20090107-2000PRD.zip` into the main folder of the Eclipse instance.

References

- [AGMO06] Ivica Aracic, Vaidas Gasiunas, Mira Mezini, and Klaus Ostermann. Overview of CaesarJ. *Transactions on AOSD I, LNCS*, 3880:135 – 173, 2006.
- [GMS⁺09] Vaidas Gasiūnas, Mira Mezini, Pablo Sánchez, Jacques Noyé, Mario Südholdt, Angel Núñez, Ludger Fiege, Iris Groher, Christine Schwanninger, Markus Völter, Steffen Zchaler, and Lidia Fuentes. Methodology for using AOP and MDD in combination for variability management in SPLs. Technical Report Deliverable D3.4, AMPLE Project, October 2009.
- [GSN⁺07] Vaidas Gasiūnas, Pablo Sánchez, Carlos Nebrera, Nadia Gámez, Lidia Fuentes, Jacques Noyé, Mario Südholdt, Angel Núñez, Christoph Pohl, Andreas Rummler, Iris Groher, Christine Schwanninger, and Markus Völter. Overview of extensions/improvements to existing implementation technologies. Technical Report Deliverable D3.2, AMPLE Project, December 2007.