Discipline(s) : Infomatique et télécommunications

THE ART OF DOMAIN-SPECIFIC LANGUAGES: LET'S HACK OUR OWN LANGUAGES!

Nature	UE

RESPONSABLES

Jean-Marc Jézéquel

OBJECTIFS

The Science of Software faces new challenges with the advent of modern software-intensive systems such as complex critical embedded systems, cyber-physical systems and Internet of things. Application domains range from robotics, transportation systems, defense to home automation, smart cities, and energy management, among others. Software is more and more pervasive, integrated into large and distributed systems, and dynamically adaptable in response to a complex and open environment. As a major consequence, the engineering of such systems involves multiple stakeholders, each with some form of domain-specific modeling.
Model-Driven Engineering (MDE) aims at reducing the accidental complexity associated with developing complex software-intensive systems through the use of modeling techniques that support separation of concerns and automated generation of system artifacts from models. Separation of concerns is founded on the exploitation of different domain-specific modeling languages (DSLs), each providing constructs based on abstractions that are specific to a concern of a system. As such, DSLs are "the heart and soul" of MDE, and have major consequences on modern development processes.
The integration of domain-specific concepts and best practices development experience into DSLs can significantly improve software and systems engineers productivity and system quality. For such a purpose, the development of DSLs has been recently recognized as a significant software engineering task itself. Indeed, the development of DSLs is a challenging task which requires specialized knowledge. This recently resulted in the emergence of Software Language Engineering (SLE), defined as the application of systematic, disciplined, and measurable approaches to the development, use, deployment, and maintenance of software languages.
This course provides an end-to-end coverage of the engineering of DSLs to turn domain knowledge into tools. It introduces the foundations of MDE and SLE, with a specific focus on the use of modeling techniques for designing and implementing DSLs. It also provides various illustrations through the definition of different kinds of DSLs, their instrumentation with tools such as editors, simulators and generators, the integration of multiple DSLs to achieve a system view, and the validation of both models and tools. Finally the course provides the foundations for engineering a family of related DSLs, for modeling and managing variability, and for synthesising billions of variants out of textual or graphical specifications.

KEYWORDS

DSL, MDE, SLE, SPL, Variability Management, Generative programming, early V&V, Separation of Concerns

PREREQUISITES

Basis of the Theory of Languages, Compilation, Object-Oriented Design, Typing

CONTENTS

The course starts with an introduction to MDE and SLE, and then moves into a deeper discussion of how to express the knowledge of particular domains into tool supported DSLs. The second part lets students investigate the applications of MDE and SLE to different types of software systems, from different starting points (language, business knowledge, standard, etc.) and for different software engineering activities such as Requirement Engineering, Variability Management, Analysis, Design, Implementation, and Validation & Verification.

Introduction to Model-Driven Engineering and Software Language Engineering
Domain-Specific Languages: Basics
Domain-Specific Languages: Advanced (staging, att. grammars, typing, op. semantics…)
Language workbenches
Program and model transformation (static analysis, code/test/doc generation…)
Program and model execution, simulation and debugging
Program and model composition (merge, coordination, synchronization)
Integration of formal methods within DSLs (e.g., model checking)
Family of DSLs and variability modeling (software product lines, feature models, etc.)
Automated reasoning (e.g., with solvers) and synthesis of variants
Cost-effective strategies for verifying billions of variants

LEARNING OUTCOMES

Design and implement families of DSL and their tooling support, generative approaches with early early V & V, application to variability management for SPL

APPARTIENT À

Master 2 informatique parcours Science Informatique

Mise à jour le 17 juillet 2017

CONTACT(S)

Département Informatique et télécommunications
École normale supérieure de RennesCampus de Ker LannAvenue Robert Schuman
35170 BRUZ
Tél. : 02 99 05 52 43
E-mail
Site Internet