DOCTORAL CANDIDATE:	Xiaorui Zhang
DEGREE:	Philosophiae Doctor
FACULTY:	Faculty of Mathematics and Natural Sciences
DEPARTMENT:	Department of Informatics
AREA OF EXPERTISE:	Model-Driven Software Engineering, Software Product Lines
SUPERVISORS:	Øystein Haugen, Birger Møller-Pedersen
DATE OF DISPUTATION:	14th of March 2014
DISSERTATION TITLE:	*Developing Model-Driven Software Product Lines*

This thesis presents a set of methodologies and automatic tool support for generating code for a family of similar softwares from software descriptions (models), in another word, model-driven software product lines, which is combination of Software Product Lines and Model-Driven Software Engineering.

What is Software Product Lines? As we know, product lines are widely used in manufacturing industry for optimized productivity: For a specific car manufacturer, all car parts are standardized, and the difference between a basic and upgraded car model might be just seat heating and built-in GPS. Software Product Lines borrow the idea: instead for producing code for individual softwares, a software family (just like a car family) is defined based on how similar and how different each family member is, and how they can be produced by assembling reusable and standardized code parts.

What is Model-Driven Software Engineering? Think about car industry again. Imagine that you are a car designer who knows all about how a perfect car should look like and function, but naturally nothing about programming. However, for the software embedded in a car, you have to tell a programmer what functionalities you need and leave the coding work to them. You can probably imagine the pain of communication and collaboration when you know nothing about programming and the programmer knows little about car specifics. With the paradigm of model-driven software engineering, a domain-specific modeling language will be developed for you beforehand, with only car-specific terms that you are familiar with. With this language, you describe how you want your car to be in great details, and then the software code for the car can be automatically generated from your description.

In particular, this thesis provides methods and tools for the complete life cycle of model-driven software development, from how to define a language for a specific domain, and how to define common and unique parts (described using the domain-specific language) among all software product members, to how to produce similar softwares by assembling reusable parts, and how to evolve a software product line over time when needed, e.g. introducing new software products into the family.