Chapter 4

Ptolemaic Metamodelling?
The Need for a Paradigm Shift

Brian Henderson-Sellers
University of Technology, Australia

Owen Eriksson
Uppsala University, Sweden

Cesar Gonzalez-Perez
Spanish National Research Council (CSIC), Spain

Pär J. Ågerfalk
Uppsala University, Sweden

ABSTRACT

By consideration of scientific paradigm shifts, in this chapter the authors evaluate possible parallels in the evolution of modelling, and particularly metamodelling and modelling language construction, as a basis for evaluating whether or not the time is ripe for a similar change of direction in model language development for software engineering. Having identified several inconsistencies and paradoxes in the current orthodoxy, they then introduce a number of ideas from outside software engineering (including language use, philosophy, and ontology engineering) that seem to solve many of these issues. Whether these new ideas, together, are sufficient to create a shift in mindset or whether they are simply the stimulus for others to create new and orthogonal ideas remains to be seen. The authors urge the modelling and metamodelling communities to search out that new orthodoxy (i.e. instigate a paradigm shift) that will, necessarily, ensure that the science will offer simpler and more satisfying solutions in the years to come.

1. INTRODUCTION

Model-Driven Engineering (MDE) focuses on models and model transformations. At the very heart of modelling is the need for models to be depicted in some way by using modelling languages (e.g. Selic, 2011), typically specified by a metamodel (e.g. Sanchez Cuadrado, de Lara, & Guerra, 2012). Metamodels are thus a (or maybe the) core element in MDE – current and future metamodelling approaches are the topic of this chapter.

The scientific approach to knowledge creation and acquisition has been the topic of many books and papers. It embodies a combination of theory building and theory testing and the ability to re-
place old theories when newer data and/or newer theories offer a more consistent and scientific explanation of the topic under consideration. The framework in which the theory resides also implies a philosophical underpinning, a paradigmatic choice that is seldom discussed or even appreciated. This interaction between theory and empirically obtained data for testing that theory is crucial in science and, hence also, in all branches of computing (e.g. Genova, 2010). Notwithstanding, there is an unfortunate trend in journal reviewing of software engineering research to reject papers that have little or no empirical support (as observed by Genova, 2010). Indeed, Broy (2011) observes that “in the long run, software engineering cannot be established without a solid body of scientific theory.” When scientific frameworks are totally reconstructed, often because they have degraded to a point where their coherence is challenged, it is said that a paradigm shift has occurred (Kuhn, 1962).

In this chapter, we focus on models and metamodels. Models have a long history in scientific research: “If you want to understand some aspect of the Universe, it helps if you simplify it as much as possible, and include only those properties and characteristics that are essential to understanding. If you want to determine how an object drops, you don’t concern yourself with whether it is new or old, is red or green, has an odour or not. You eliminate those things and thus do not needlessly complicate matters. The simplification you can call a model or a simulation and you can present it either as an actual representation on a computer screen, or as a mathematical relationship” (Asimov, 1988). However, even with this illustrious and lengthy history, Ritchey (2012) argues that there is still no general theory of modelling.

Here, we take a largely theoretical approach, where our evaluation approach uses examples taken from considering the real world and asking whether each model provides a good or a better appreciation of the target problem domain. In order to create such a comprehensive modelling approach, we integrate knowledge from various sub-fields of computing: software engineering, modelling (especially the use of UML and, more generally, conceptual modelling), as well as considering research knowledge from outside the traditional context of computing: ontology engineering, philosophy, mathematics, language use (especially speech acts) and cognitive linguistics. For the non-computing fields, we note that each has its own conferences and journals and, importantly, their own communities, which tend often to be ‘clubs’ or ‘closed shops’ with their own internal terminology immediately understandable by members of the club but unintelligible to those outside (e.g. Kuhn, 1962). To improve our models and our modelling axioms and constraints, we need to break into each and every of these clubs.

In this chapter, we will argue that the time is ripe for a paradigm shift in software language engineering and in particular in the model we use to understand multilevel metamodel architectures, especially relevant for MDE. Firstly, we will outline a parallel in the history of science: that of the Copernican Revolution (Section 2.1). In Section 2.2, we will give a brief overview of the history of software engineering metamodelling. Whilst Section 2 describes some of the problems and some of the fixes made at a high level, Section 3 delves into the detail of each of these proposed solutions. In particular, we explore the generalization\(^1\) and instantiation relationships, the potential value of powertypes, and how ontology engineering and language use can add value to the creation and use of modelling languages. Despite these new additions to software language engineering, problems remain; in Section 4 we propose that the time is ripe for a minor revolution in thinking with respect to the relative positioning of metamodels, modelling languages and multilevel frameworks.