An Ontology-Based and Model-Driven Approach for Designing IT Service Management Systems

Maria-Cruz Valiente, University of Alcalá, Spain

Cristina Vicente-Chicote, Technical University of Cartagena, Spain

Daniel Rodríguez, University of Alcalá, Spain

ABSTRACT

Currently, few projects applying a Model-Driven Engineering (MDE) approach start from high-level requirements models defined exclusively in terms of domain knowledge and business logic. Ontology Engineering (OE) aims to formalize and make explicit the knowledge related to a particular domain. In this vein, this paper presents a modeling approach, formalized in ontological terms, for defining high-level requirements models of software systems that provide support for the implementation of Information Technology Service Management Systems (ITSMs). This approach allows for: (1) formalizing the knowledge associated to the ITSM processes contained in an ITSM; (2) modeling the semantics of the activities associated to these processes in terms of workflows; (3) automatically generating the high-level requirements models of the workflow-based software systems needed to support (part of) the ITSM processes; and (4) from the latter, obtaining lower-level models (and eventually code) by means of automated model transformations. A real case study describing the use of this proposal to model an Incident Management System is also included to demonstrate the feasibility and the benefits of the proposed approach.

Keywords: Business Management, Incident Management Systems, IT Service Management, Knowledge Management, Model-Driven Engineering

1. INTRODUCTION

In this day and age no one can question the importance of Information Technology (IT) in the business world. A world in which there is a market that is more and more competitive. With the continuous integration and standardization of new computer technologies, the business world is changing frequently. Therefore the business world is immersed in a cycle of continuous improvement, where essentially, the level of quality of the IT services delivered to the customers is often the deciding and differentiating factor. Thus, business people
have increased their expectations related to
the IT department, and now they need IT to
support their business processes in a strategic
way. That is, organizations are aware of the
closer relationship and convergence between
business and IT.

In this vein, IT Service Management
(ITSM) provides a set of specialized organiza-
tional capabilities and a professional practice,
supported by an extensive body of knowledge,
experience and skills for providing value to cus-
tomers in the form of IT services (OGC, 2007).

The implementation of any IT service-
oriented software system requires performing
a number of different steps in order to produce
all the required artifacts (either internal or de-

deliverable). Based on the notion that a software
system is a representation of another system
(i.e., the real-world), the first step is to formal-
ize the domain concepts and the relationships
between them (i.e., the ontology), in order to
obtain a common vocabulary agreed by all
the stakeholders involved in a given project
for requirements elicitation. In addition, apart
from the domain concepts, additional rules,
constraints and semantics are required in order
to avoid semantic ambiguities, uncertainties and
contradictions. The Web Ontology Language
(OWL) (Smith, Welty, & McGuinness, 2004),
the de facto standard for ontology representa-
tion, enables the definition of rules, constraints
and semantics in terms of logic based domain
concepts. The importance of an investigation
of the issues involved in the IT service-oriented
requirements analysis is also remarked by Lich-
tenstein, Nguyen & Hunter (2004). However, in
spite of the efforts of the Software Engineer-
ing (SE) community to define new intuitive and
powerful techniques, there is still an open gap
regarding the automated and seamless integra-
tion of domain aspects (i.e., the business view)
into the software development process.

The emerging Model-Driven Engineering
(MDE) paradigm offers a promising solution
to cope with this limitation. MDE addresses
the inability of third-generation languages to
cope with increasing software complexity,
allowing us to describe domain concepts ef-
fectively (Schmidt, 2006; Gašević & Hatala,
2010). Model Driven Engineering (MDE) is
a software and system construction approach
based on high-level abstract modeling. All the
relevant information in a project is stored in
models based on well-defined languages and
development is then carried out as a sequence
of model transformations. The MDE term was
first proposed by Kent (2002) but it is derived
from the OMG’s Model Driven Architecture
(MDA) initiative (OMG, 2003).

As shown in Figure 1, MDA defines a
particular MDE process aimed at separating the
business logic from the technological platforms.
Thus, organizations can use MDA to meet the
integration challenges posed by new platforms,
while preserving their investments in existing
business logic. MDA proposes three modeling
levels, namely (ordered from highest to lowest
levels of abstraction): Computation Independent
Model (CIM), Platform Independent Model
(PIM), and Platform Specific Model (PSM). Dif-
ferent Model-to-Model (M2M) transformations
among these abstraction levels can be defined
either top-down or bottom-up. Commonly,
each CIM (model gathering high-level busi-
ness requirements, sometimes called a domain
model) is transformed into one or more PIMs
(platform-independent architectural models).
Similarly, each PIM is transformed into one or
more PSMs (one for each target platform). PSMs
are commonly very low level models, enabling
the definition of a direct Model-to-Text (M2T)
transformation for automatically generating the
final system implementation, including code,
documentation, etc.

At the model layer (M1), CIMs are com-
monly high-level business models that represent
the high-level requirements for the system to
build (M0). A high-level requirement is focused
on the actual stakeholders problems and needs
and describes the characteristics of the domain
of the systems (that is, what is needed, but not
how this is to be implemented) (Olivé, 2007).
Therefore, CIMs help in bridging the gap be-
between the conceptual level mainly performed
by domain experts and the implementation
level performed by the designers and develop-
Related Content

The AHP Method for the Evaluation and Selection of Complementary Products

Modeling and Governance of Procurement as a Service Responsive to Business Events
[www.igi-global.com/chapter/modeling-governance-procurement-service-responsive/50244?camid=4v1a](www.igi-global.com/chapter/modeling-governance-procurement-service-responsive/50244?camid=4v1a)
Business Transformation Workbench: A Practitioner’s Tool for Business Transformation
Juhnyoung Lee, Rama Akkiraju, Chun Hua Tian, Shun Jiang, Rong Zeng Cao, Siva Danturthy, Ponn Sundhararajan, Rakesh Mohan, Wei Ding and Carl Nordman (2011). Service Intelligence and Service Science: Evolutionary Technologies and Challenges (pp. 130-171).
www.igi-global.com/chapter/business-transformation-workbench/47360?camid=4v1a

A Novel Security Framework for Managing Android Permissions Using Blockchain Technology
www.igi-global.com/article/a-novel-security-framework-for-managing-android-permissions-using-blockchain-technology/196191?camid=4v1a