Representing Micro-Business Requirements Patterns with Associated Software Components

RJ Macasaet, Department of Software Research and Development, Pentathlon Systems Resources Inc., Manila, Philippines

Manuel Noguera, Department of Computer Languages and Systems, University of Granada, Granada, Spain

María Luisa Rodríguez, Department of Computer Languages and Systems, University of Granada, Granada, Spain

José Luis Garrido, Department of Computer Languages and Systems, University of Granada, Granada, Spain

Sam Supakkul, Applications Architecture, Sabre Inc., Southlake, TX, USA

Lawrence Chung, Department of Computer Science, University of Texas at Dallas, Richardson, TX, USA

ABSTRACT

This paper proposes representations for micro-business requirements patterns with associated software components. The patterns must be comprehensible enough for micro-business owners without technical backgrounds but at the same time be technical enough for the software developers who use them during the design and development of software. Both objectives are somewhat conflicting and trade-offs have to be made regarding their representations. The pattern representations use a combination of tables, business process models, goal graphs, labels, and UML component diagrams. First, the paper provides an example of a micro-business requirements pattern in the form of an inventory pattern and applies it in a real-world micro-business case, a clothes retail store. Through the example, it shows how the pattern is instantiated and associated with software components. Then, it shows how the patterns are applied in industrial practice, including the software development companies currently adapting and applying them, accompanied by observable strengths and weaknesses.

Keywords: Business Process Modeling, Component-Based Development, Functional Requirements, Micro-Business, Non-Functional Requirements, Requirements Pattern, Reusable Software, Softgoal Interdependency Graphs, Software Components, UML

DOI: 10.4018/ijismd.2014100104
1. INTRODUCTION

A micro-business is the smallest small-to-medium sized enterprise (SME). Its domain is complex, filled with several challenges, and is normally not studied in academia (Kelliher & Reinl, 2009). The simple question, “what is a micro-business?” already has its fair share of arguments. (European Commission, 2008; International Organization for Standardization, 2011) characterize (micro-)businesses based on the number of employees (ISO refers to micro-businesses as very small entities or VSEs). However, several researchers believe otherwise (Merten et al., 2011). Nikula et al. (2000) argue that the age of a (micro-)business is a better metric.

In relation to software for micro-businesses, (Jantunen, 2010) argues that the degree of collaboration in software projects should be the basis for defining a (micro-)business. (Kamsties et al., 1998) say that the adaptability of (micro-)businesses in software projects is a better metric. (Aranda et al., 2007; Aranda, 2010) say that the length of software projects should characterize (micro-)businesses and this description would perhaps be the closest to ours.

We characterize a micro-business based on the number of software components required in its prospective software system. The implementation of software systems requiring more than a small number of components (based on our industry experience, we made an informal estimate to be about 10 or so (Macasaet et al., 2012)) could increase the complexity of a project and the length of the software project (or man days required to complete the project), resulting in cost estimates that are out of range for micro-businesses. There would definitely be some exceptions to this estimation but these exceptions would be out of the norm.

We characterize software components as having independent modules encapsulating a certain set of data and functions, varying in granularity as long as they may be updated, replaced, or modified without affecting other software components. Examples of software components include, but are not limited to, an off-the-shelf accounting system, a website template, a customer database, or even a simple cascading style sheet (CSS) file. A similar characterization of components is made by Medvidovic and Taylor (2000) where components are defined as units of computation or data store which could be as small as a single procedure or as large as an entire application.

Micro-business owners are busy people, dedicated to improving their profit bottom lines on a daily basis. They are a proud bunch too, reluctant to learn technical software jargon (Kamsties et al., 1998; Kauppinen et al., 2002) and are comfortable when using their natural language and sketches to express themselves to software developers (Macasaet et al., 2011). This kind of behavior makes the communication between micro-business owners and software developers an interesting affair.

Micro-businesses need software to improve their competitiveness in the market, be it a customer relationship management (CRM) system, a point-of-sale (POS) system, or even a micro-scale enterprise resource planning (ERP) system. The communication of software requirements is an important part of software, regardless of the size of the business (Young, 2004). If the software requirements are not done properly then there will eventually be problems during the implementation (Kauppinen et al., 2004) and acceptance of the software, threatening the overall success of the project (Davis et al., 2006).

If micro-business owners are not inclined to communicate their software requirements using technical software terms then the communication of requirements must be done as intuitively and as comprehensibly as possible (Kruchten, 2003), with little or no technical terms involved (Young, 2004). This kind of requirements approach for micro-businesses should have a “lightweight but effective” (Ambler, 2002) tone.

However, software developers need requirements information which is technically relevant and which would enable them to be more competitive in doing their work as well. The requirements are the blueprint for the
Related Content

Multi-Modeling, Meta-Modeling, and Workflow Languages
[www.igi-global.com/chapter/multi-modeling-meta-modeling-and-workflow-languages/91941?camid=4v1a](www.igi-global.com/chapter/multi-modeling-meta-modeling-and-workflow-languages/91941?camid=4v1a)

Quality-Driven Model Transformations: From Requirements to UML Class Diagrams
[www.igi-global.com/chapter/quality-driven-model-transformations/26834?camid=4v1a](www.igi-global.com/chapter/quality-driven-model-transformations/26834?camid=4v1a)
Execution Management for Mobile Service-Oriented Environments
www.igi-global.com/article/execution-management-mobile-service-oriented/47037?camid=4v1a

Rule-Based Domain-Specific Modeling for E-Government Service Transactions
www.igi-global.com/chapter/rule-based-domain-specific-modeling-for-e-government-service-transactions/117929?camid=4v1a