Chapter XVI
Some Method Fragments for Agile Software Development

Q.N.N. Tran
University of Technology, Sydney, Australia

B. Henderson-Sellers
University of Technology, Sydney, Australia

I. Hawryszkiewycz
University of Technology, Sydney, Australia

ABSTRACT

The use of a situational method engineering approach to create agile methodologies is demonstrated. Although existing method bases are shown to be deficient, we take one of these (that of the OPEN Process Framework) and propose additional method fragments specific to agile methodologies. These are derived from a study of several of the existing agile methods, each fragment being created from the relevant powertype pattern as standardized in the Australian Standard methodology metamodel of AS 4651.

INTRODUCTION

It is increasingly recognized that a universally applicable methodology (a.k.a. method) for software (and systems) development is not possible (Brooks, 1987; Avison & Wood-Harper, 1991; Fitzgerald, Russo, & O’Kane, 2003). One way to approach this is to eschew all attempts to create and promote a single methodology but instead to create a repository (or methodbase: Saeki, Iguchi, Wen-yin, & Shinohara, 1993) containing a large number of method fragments gleaned from a study of other methodologies, an evaluation of best industry practice, and so forth. Situational methods (Kumar & Welke, 1992; Odell, 1995) are then constructed by a method engineer “bottom up” from these fragments in such a way that they are “tailored” to the process requirements of the industry in question. This is the method engineering (ME) or situational method engineering (SME) approach to methodologies.

A second thread of relevance is the increasing interest, both in academe and industry, of agile
Some Method Fragments for Agile Software Development

methods—methodological approaches to software
development that tend to the minimalistic, focus on
people rather than documented processes, and react
well to rapidly changing requirements (Abrahams-
son, Warsta, Siponen, & Ronkainen, 2003; Turk,
France, & Rumpe, 2005). However, as published and
often as practiced, these agile methods themselves
may be overly rigid. To make them more flexible and
possess so-called “dual agility” (Henderson-Sellers
& Serour, 2005), a method engineering approach
can be applied to agile methods as well as more
traditional software development approaches. To
do so, it is incumbent upon the method engineers
who provide the method bases to ensure that these
repositories of method fragments contain adequate
fragments from which a range of agile methods can
indeed be constructed.

In this chapter, we hypothesize that an agile
method can be created from method fragments, once
those fragments have been identified and appropri-
cately documented. Following an introduction to the
general characteristics of agile software develop-
ment, we then examine an underpinning metamodel
(AS4651). We then identify and document method
fragments that conform to this metamodel and
that support a range of agile methods including
XP, Crystal, Scrum, ASD, SDSM, and FDD. We
thus propose the addition of these newly document
fragments to one extensive ME repository, that of
the OPEN Process Framework (OPF) (Firesmith
& Henderson-Sellers, 2002; http://www.opfro.org),
chosen on the basis of it having the most extensive
content in its methodbase. An important part of
any such research is the validation phase. This is
described in the complementary chapter (Tran, Hen-
derson-Sellers, & Hawryszkiewycz, 2007), where
we (re-)create four agile methods from the fragments
in the newly enhanced OPF methodbase.

GENERAL CHARACTERISTICS OF
AGILE SOFTWARE DEVELOPMENT

Although each agile development methodology is
distinct, they do share some common characteris-
tics. Agile development adheres to the following
fundamental values (Agile Manifesto, 2001):

• **Individuals and interactions** should be more
  important than processes and tools.
• **Working software** should be more important
  than comprehensive documentation.
• **Customer collaboration** should be more im-
  portant than contract negotiation.
• **Responding to change** should be more im-
  portant than following a plan.

Firstly, agile development emphasizes the rela-
tionship and communality of software developers,
as opposed to institutionalized processes and develop-
ment tools. Valuing people over processes allows
for more creativity in solutions. In the existing agile
practices, this value manifests itself in **close team
relationships, close working environment arrange-
ments**, and other **procedures boosting team spirit**.
The importance of teamwork to agile development
has been emphasized by agilists (Cockburn & High-

Secondly, an important objective of the software
team is to continuously produce tested working
software. It is argued that documentation, while
valuable, takes time to write and maintain, and
is less valuable than a working product. Some
agile methodologies promote **prototyping** (e.g.,
ASD), while others encourage **building simple but
completely functional products quickly** as possible
(e.g., XP).

Thirdly, **customer involvement** is promoted in
all agile methodologies. The relationship and co-
operation between the developers and the clients
are given the preference over strict contracts. The
clients are encouraged to actively participate in the
development effort.

Fourthly, the developers must be prepared to
**make changes** in response to the emerging/chang-
ing needs during the development process. Any
**plan must be lightweight** and easily modifiable. The
“plan” might simply be a set of post-it notes on a
whiteboard (e.g., as in Scrum: Schwaber, 1995).
Related Content

IDS Using Reinforcement Learning Automata for Preserving Security in Cloud Environment
[www.igi-global.com/article/ids-using-reinforcement-learning-automata-for-preserving-security-in-cloud-environment/205594?camid=4v1a](www.igi-global.com/article/ids-using-reinforcement-learning-automata-for-preserving-security-in-cloud-environment/205594?camid=4v1a)

SLIM: Service Location and Invocation Middleware for Mobile Wireless Sensor and Actuator Networks
[www.igi-global.com/article/slim-service-location-invocation-middleware/47038?camid=4v1a](www.igi-global.com/article/slim-service-location-invocation-middleware/47038?camid=4v1a)

An Empirical Investigation into the Adoption of Open Source Software in Hospitals
[www.igi-global.com/chapter/empirical-investigation-into-adopter-open/29467?camid=4v1a](www.igi-global.com/chapter/empirical-investigation-into-adopter-open/29467?camid=4v1a)

Requirements Risk and Maintainability
[www.igi-global.com/chapter/requirements-risk-maintainability/4903?camid=4v1a](www.igi-global.com/chapter/requirements-risk-maintainability/4903?camid=4v1a)