The Essentials of Modern Software Engineering
*Free the Practices from the Method Prisons!*

by Ivar Jacobson, Harold “Bud” Lawson, Pan-Wei Ng, Paul E. McMahon, and Michael Goedicke

**Table of Contents**

Abstract

Foreword by Ian Sommerville

Foreword by Grady Booch

Preface

Part 1 – The Essence of Software Engineering

1 From Programming to Software Engineering
   1.1 Beginning with Programming
   1.2 Programming is Not Software Engineering
   1.3 From Internship to Industry
   1.4 Journey into the Software Engineering Profession

2 Software Engineering Methods and Practices
   2.1 Software Engineering Challenges
   2.2 The Rise of Software Engineering Methods and Practices
      2.2.1 There are Lifecycles
      2.2.2 There are Technical Practices
      2.2.3 There are People Practices
      2.2.4 Consequences
   2.3 The SEMAT Initiative
   2.4 Essence: The OMG Standard

3 Essence in a Nutshell
   3.1 The Ideas
   3.2 Methods are Compositions of Practices
   3.3 There is a Common Ground
   3.4 Focus on the Essentials
   3.5 Provide an Engaging User Experience
4 Identifying the Key Elements of Software Engineering

4.1 Getting to the Basics

4.2 Software Engineering is About Delivering Value to Customers
   4.2.1 Opportunity
   4.2.2 Stakeholders

4.3 Software Engineering Delivers Value Through a Solution
   4.3.1 Requirements
   4.3.2 Software System

4.4 Software Engineering is Also About Endeavors
   4.4.1 Team
   4.4.2 Work
   4.4.3 Way of Working

5 The Language of Software Engineering

5.1 A Simple Practice Example

5.2 The Things to Work With
   5.2.1 Alphas
   5.2.2 Alpha States
   5.2.3 Work Products

5.3 Competencies

5.4 Things to Do
   5.4.1 Activities

5.5 Essentializing Practices

6 The Kernel of Software Engineering

6.1 Organizing Essence Kernel

6.2 The Essential Things to Work With: The Alphas

6.3 The Essential Things to Do: The Activities

6.4 Competencies

6.5 Patterns
   6.5.1 Roles
6.5.2 The Checkpoint Pattern

7 Key Principles: Practical, Actionable, and Extensible

7.1 The Big Idea
7.2 Essence is Practical
7.3 Essence is Actionable
7.4 The Kernel is Extensible
7.5 How Does the Kernel Relate to Agile and Other Existing Approaches?
7.6 How Essence Will Help You

8 Reflection on Theory

8.1 Where’s the Theory for Software Engineering?
8.1.1 What is Software Engineering Theory?
8.1.2 Three Arguments
8.2 Uses of Theory
8.3 The Essence is a General, Descriptive Theory of Software Engineering
8.4 Towards a General Predictive Theory of Software Engineering
8.5 A Theoretical Foundation Helps You Grow

Part 2 – Developing Software with Essence

9 Applying Essence in the Small – Playing Serious Games

9.1 Progress Poker
9.1.1 Benefits of Progress Poker
9.1.2 Example of Travel Essence Team Playing Progress Poker
9.2 Chasing the State
9.3 Objective Go
9.4 Checkpoint Construction
9.5 Reflection

10 Kick Starting Development Using Essence

10.1 Understand the Context Through the Lens of Essence
10.2 Agreeing on the Development Scope and Checkpoints
10.3 Agreeing on the Most Important Things to Watch
11 Developing with Essence
  11.1 Planning with Essence
  11.2 Doing and Checking with Essence
  11.3 Adapting a Team’s Way of Working with Essence
  11.4 How the Kernel Helps You in Adapting the Way of Working
    11.4.1 Helping a Team Reason About Their Way of Working
    11.4.2 Making Changes to the Way of Working

12 The Development Journey
  12.1 Visualizing the Journey
  12.2 Ensuring Progress and Health
  12.3 Dealing with Anomalies

13 Reflection on Kernel
  13.1 Validity of the Kernel
  13.2 Applying the Kernel Effectively

Part 3 – Small Scale Development with Practices

14 Kick Starting Development with Practices
  14.1 Understand the Context Through the Lens of Essence
  14.2 Agree Upon Development Scope and Checkpoints
  14.3 Agree upon Practices to Apply
  14.4 Agree upon the Important Things to Watch
  14.5 Journey in Brief

15 Running with Scrum
  15.1 Scrum Explained
  15.2 Practices Make a Software Engineering Approach Explicit and Modular
  15.3 Making Scrum Explicit Using Essence
  15.4 Elements of the Scrum Lite Practice
    15.4.1 Scrum Lite Alphas
  15.5 Scrum Lite Work Products
    15.5.1 Product Backlog
15.5.2 Sprint Backlog
15.5.3 Increment

15.6 Scrum Lite Roles

15.7 Kick Starting Scrum Lite Usage

15.8 Working with Scrum Lite
   15.8.1 Sprint Planning
   15.8.2 Daily Scrum
   15.8.3 Sprint Review
   15.8.4 Sprint Retrospective

15.9 Reflecting on the Use of Scrum with Essence
   15.9.1 Adding Explicit Practices
   15.9.2 Visualizing the Impact of the Scrum Lite Practice
   15.9.3 Value of Being Precise

16 Running with User Story Lite
   16.1 User Stories Explained
   16.2 Making the User Story Lite Practice Explicit Using Essence
   16.3 User Story Lite
   16.4 User Story Lite alphas
      16.4.1 User Story
   16.5 User Story Lite Work Products
      16.5.1 Story Card
      16.5.2 Test Case
   16.6 Kick Starting User Story Lite Usage
   16.7 Working with User Story Practice
      16.7.1 Find User Stories
      16.7.2 Prepare a User Story
      16.7.3 Applying the Splitting User Stories Pattern
      16.7.4 Accept a Story
   16.8 The Value of the Kernel to the User Story Practice
16.8.1 Visualizing the Impact of the User Story Practice

17 Running with Use-Case Lite

17.1 Use Cases Explained

17.1.1 Use-Case Lite

17.2 Making the Use-Case Lite Practice Explicit Using Essence

17.3 Use Case Lite Alphas

17.3.1 Progressing Use Cases

17.3.2 Progressing Use Case Slices

17.4 Use Case Lite Work Products

17.4.1 Use-Case Model

17.4.2 Use Case Narrative

17.4.3 Use-Case Slice Test Case

17.5 Kick Starting Use Cases to Solve a Problem Our Team is Facing

17.5.1 Find Actors and Use Cases

17.6 Working with Use Cases and Use-Case Slices

17.6.1 Slice the Use Cases

17.6.2 Prepare a Use Case Slice

17.6.3 Test a Use Case Slice

17.7 Visualizing the Impact of Using Use Cases for the Team

17.8 Progress and Health of Use Case Slices

18 Running with Microservices

18.1 Microservices Explained

18.2 Making the Microservice Practice Explicit Using Essence

18.3 Microservices Lite

18.4 Microservices Lite alphas

18.5 Microservices Lite Work Products

18.5.1 A Brief UML Primer

18.5.2 Design Model

18.5.3 Microservice Design
22.3.3 Agree on Practices to Apply
22.3.4 Agree on the Important Things to Watch
22.3.5 Agreeing What to Develop
22.3.6 Managing the Work
22.3.7 Working Iteratively
22.3.8 Continual Improvement

22.4 Value of Essence to Large Scale Development

23 Reaching Out to Different Kinds of Development

23.1 From A Practice Architecture to a Method Architecture
23.2 Establishing a Practice Library Within an Organization

24 Being Agile with Practices and Methods

24.1 Agile Manifesto
24.2 Agile Working with Methods
24.3 The Full Team Owns Their Method
24.4 Focus on Method Use
24.5 Evolve Your Team's Method

25 Reflecting on the SEMAT Journey

APPENDIX: A Brief History of Software and Software Engineering