Foreward to Handbook of Research on Innovations in Systems and Software Engineering

Citation for published version (APA):
Zhao, L., Díaz, V. G. (Ed.), Lovelle, J. M. C. (Ed.), & García-Bustelo, B. C. P. (Ed.) (2014). Foreward to Handbook of Research on Innovations in Systems and Software Engineering. In *Handbook of Research on Innovations in Systems and Software Engineering* IGI Press.

Published in:
Handbook of Research on Innovations in Systems and Software Engineering

Foreword

While traditionally taught as two separate subjects, Systems Engineering (SYE) and Software Engineering (SE) have become increasingly intertwined, with SYE methods widely adapted to SE and SE methods to SYE (Fairly & Willshire, 2011), as Table 1 shows. This complementarity between SYE and SE has naturally given rise to an interdisciplinary approach called "Systems and Software Engineering (SSE)," which integrates successful methods and best practices in both SYE and SE and offers a holistic approach to systems development.

The question thus arises: Why do we need such an approach? There are at least two answers.

First, today's systems comprise not only hardware, but also software. The division between the physical and digital worlds has become blurring (Feiler, et al., 2006). Therefore, traditional SE approaches or SYE approaches are no longer adequate to today's complex systems development.

Second, from the software development perspective, there are unprecedented demands on high quality software and low failure rates in software and software projects (Standish Group, 2003). The inability of current SE practices to meet these demands calls for new approaches. SSE is a promising approach as it combines best practices of both SE and SYE.

Yet, as a new approach, we have a limited understanding as to how it will make a difference in future systems development and what tools and techniques it affords. The *Handbook of Research on Innovations in Systems and Software Engineering* provides this knowledge. The Handbook offers an excellent sampling of current research and development in SSE and presents advanced SSE tools and techniques for building successful software. Specifically, the book contains 28 solid chapters that cover three critical

Table 1. Cross-fertilisation between systems engineering and software engineering (Fairly & Willshire, 2011)

Systems Engineering Methods Adapted to Software Engineering	**Software Engineering Methods Adapted to Systems Engineering**
• Stakeholder Analysis. • Requirements Engineering. • Functional Decomposition. • Design Constraints. • Architectural Design. • Design Criteria. • Design Trade-offs. • Interface Specification. • Traceability. • Configuration Management. • Systematic Verification and Validation.	• Model-Driven Development. • Software Modelling. • Use Cases. • Object-Oriented Design. • Iterative Development. • Agile Methods. • Continuous Integration. • Process Modelling. • Process Improvement. • Incremental Verification and Validation.

areas of SSE, concerning software development processes, methods and applications. The chapters on software development processes addresses issues related to software qualities, such as security, costs, and performance metrics. The chapters on software development methods focus on Model-Driven Engineering (MDE), an important approach for both SYS and SE, as Table 1 shows. Finally, the chapters on software applications are related to Mobile Software Engineering. These applications range from developing mobile environments to novel architectures to quality of services. They serve as perfect examples to elucidate why we need SSE to solve emerging software development problems. The handbook is a timely reference for researchers, practitioners and students who are interested in SSE.

Liping Zhao
University of Manchester, UK

Liping Zhao *is a senior lecturer in the School of Computer Science at the University of Manchester. She works in the area of software engineering, with a particular focus on discovering and developing reusable software patterns and finding technologies to bridge the gap between natural language specifications of software requirements and initial software models. She published the first papers on domain-specific patterns for software analysis and design and was the first to use the scientific concepts of symmetry and symmetry breaking to explain the meaning of software patterns and why they are important. Most recently, she led the research and development of an innovative software system that applies advanced natural language processing and modeling techniques to automatically transforming natural language specifications of software requirements into initial software models. She received three IBM Faculty Awards for her outstanding contributions to software patterns and service sciences.*

REFERENCES

Fairley, R., & Willshire, M. J. (2011). Teaching software engineering to undergraduate system engineering students. In *Proceedings of the 24th IEEE-CS Conference on Software Engineering Education and Training* (CSEE&T 2011) (pp. 219-226). Honolulu, HI: IEEE.

Feiler, P. H., Sullivan, K., Wallnau, K. C., Gabriel, R. P., Goodenough, J. B., Linger, R. C., & Schmidt, D. (2006). *Ultra-large-scale systems: The software challenge of future*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.

Standish Group. (2003). *The chaos report*. Author.