This book delineates a new creation process and provides an understanding of software pattern languages and true domain analysis, based on the fundamental concepts of software stability. It introduces a very well defined paradigm for creating pattern languages, software patterns, and better software development methodology that leads to highly reusable artifacts and high quality, cost-effective systems. Each chapter of the book concludes with an open research issue, review questions, exercises, and a series of projects.
Knowledge Maps (KM) enable us to do a domain analysis related to any problem domain in a way this is not tied to a specific case or a specific condition, rather the domain analysis is done in a holistic and conceptual sense. The core knowledge related to any domain is recorded, thus enabling comprehensive requirement analysis. The benefits do not stop there. KM’s properties like intersection of different KMs, using remote KMs in association with the KM of any one domain under consideration lets us analyze the knowledge across various such Knowledge maps and thus multiple domains, as a whole and help us in coming up with requirements and design of any system that spans across more than one domains of application.

Any software that is architected and designed from the knowledge captured by Knowledge Maps, is done in a way that makes it highly adaptable in nature i.e. separation of concerns between functionality sets is extremely well done. Each part of the functionality can be taken out or some other can be plugged into the architecture, scaling it in whatever way it is required. A Knowledge Map lets us come up with a stable core software system rather quickly which consists of the core knowledge for one or more applications as and how any application context is brought into the system. With the ability to design a generic stable core that can be reused in multiple scenarios, enables it to be suited and adapted to a change in the application context by changing just the application context specific parts (which are relatively a small percentage of the total system) of the software. All of this is possible since it is built on the foundation of Fayad’s Software Stability Model[]. This brings along a number of added values, such as longevity, high returns on investments, configurability, customizability, high reusability of the artifacts developed and much more.

You can check this book on Amazon

http://tinyurl.com/SoftwarePatterns

Publisher Websites:

http://www.taylorandfrancis.com/books/details/9781466571433/
http://www.crcpress.com/product/isbn/9781466571433

Other Websites:

The publisher's website
http://www.taylorandfrancis.com/books/details/9781466571433/
http://www.crcpress.com/product/isbn/9781466571433

All other websites

1. http://books.google.com/books/about/Software_Patterns_Knowledge_Ma
ps_and_Dom.html?id=Wp5uLwEACAAJ

If you like to collaborate, please join us at http://www.engr.sjsu.edu/fayad/FayadsResearchGroup/index.html

Other Links:

http://www.engr.sjsu.edu/fayad

http://www.facebook.com/drmfayad

https://www.facebook.com/fayadfoundations

https://www.researchgate.net/profile/Mohamed_Fayad/

http://twitter.com/#!/mefayad

http://www.linkedin.com/in/mefayad

http://www.youtube.com/mefayad1

http://www.engr.sjsu.edu/fayad/Egyptian%20Abroad.php

http://www.engr.sjsu.edu/fayad/Software%20Industry%20in%20Egypt.php

http://www.engr.sjsu.edu/fayad/Rules%20of%20Honor.php

http://www.engr.sjsu.edu/fayad/Government_Priorities/index.html