The software industry has evolved toward complex supplier-user networks that cooperate and collaborate in many ways. We hardly see any longer the traditional way of software development in which one company handles design, production, sales, delivery, and service. Business models, engineering life cycles, distribution channels, and services have changed dramatically. A key driver in these new value networks is open source software (OSS). Worldwide companies in various industries have invested in open source. Market leaders such as Google, IBM, Microsoft, SAP, and Siemens as well as many small companies turn to OSS for multiple reasons. For instance, Mozilla’s Firefox browser, with 150 million users, rivals Microsoft’s market-leading Internet Explorer and thus has created new momentum in an otherwise stagnating market.

Open source refers to software that we may freely use, modify, or distribute, provided we observe certain restrictions with respect to copyright and protection of its open source status. OSS is not freeware and generally has a copyright. IEEE Software has assembled this minitheme to provide a glimpse of where OSS is heading.

OSS’s Contributions to Today’s Technologies
Many major technology breakthroughs are based on OSS, and many current engineering processes have evolved from the way open source is developed. This holds for iterative development and agile techniques as well as globally distributed software engineering. The reasons are manifold.

First, global competition and low entry thresholds drive companies to continuously look for ways to reduce the cost of their software products and components. OSS with comparatively low license and maintenance costs foster life-cycle cost reduction. Second, time to profit means cutting out whatever delays the introduction of products and services. Using mature, standard OSS components lets companies focus on the high end, where true value is created. Third, practically all industries are shifting—at different speeds and cycle times—from hardware to software and finally to services. This implies that standardized components and solutions are replacing traditional proprietary software.

Also, global markets’ growing instability drives users to select endurable solutions that cannot be impacted by fads and hype, which often drive commercial software into unnecessary complexity to generate annual revenue streams. OSS delivers only core features and thus achieves better performance and quality.

Another reason is that OSS solutions with sufficiently big communities have better quality than their commercial counterparts. For instance, OSS improves security because more people review OSS source code than that of proprietary software. For this reason, security breaches are typically fixed more quickly and with broad notification to the user community.

Students are using open source in school, which substantially shortens their learning curve when they go to work for software companies. Engineers often have the same OSS tools at home, and this positively affects their office climate and productivity.

Finally, having shaped the software industry for a decade, OSS now has strong ecosystems covering embedded software components, middleware, enterprise software, Internet services, and other technologies.

Today, a variety of business models around open source are being exploited, the risks are known, and mitigating solutions exist. Not using OSS someday could mean they are unable to compete. Many OSS solutions lead their specific markets after a short time owing to the obvious benefits of such rich and
dynamic ecosystems. Specific communities have been created with suppliers and their customers using open source processes and mechanisms to provide faster access to hardware drivers, software updates, or specific features. New value networks are enhancing traditional approaches. Suppliers are teaming up to share their software basis and thus offer tailored services to single-user segments. Independent software vendors distribute popular OSS solutions and components or integrate them into other products.

The Articles
This minitheme offers two articles that both address OSS processes and tools. “Open Collaboration within Corporations Using Software Forges” looks into how to use the forge approach in internal OSS settings, called community source. Traditionally, a forge is the workplace of a smith, such as a blacksmith. A software forge is a tools platform for collaborative software development. A prominent example that you’ve all used is SourceForge.net. Many companies are using tools such as GForge to set up community source. The authors Dirk Riehle, John Ellenberger, Tamir Menahem, Boris Mikhailovski, Yuri Natchetoi, Barak Naveh, and Thomas Odenwald present software giant SAP’s experiences using a software forge to bring OSS best practices into SAP workplaces. Close to 1,000 engineers are using the SAP Forge to maintain some 200 projects. Software forges mobilize energy where top-down resource planning wouldn’t work, either because of size or conflict of departmental interests. SAP sees such voluntary work as more productive because of the “bottom-up” contribution of expertise and energy from all across the organization.

The second article, “Enhancing Open Source Software in Alignment with CMMI-DEV,” by Christiane Gresse von Wangenheim, Jean Carlo R. Hauck, and Aldo von Wangenheim, looks at how companies can benefit from OSS project management tools. They compare the most popular OSS project management applications and evaluate compliance to CMMI-DEV (CMMI for Development). They chose the CMMI because it’s today’s major industry reference for improving engineering processes. Project management is a CMMI focus because of the high correlation of project failures and insufficient project management capabilities. The authors introduce the dotProject tool and show how they enhanced its features to provide systematic project management support for small and medium-sized companies.

Both articles show how OSS mechanisms can be used to energize people and improve internal engineering and management processes. Clearly, not all software can or should be built with OSS components. Most OSS supply networks are too weak in staffing, competencies, and global reach to ensure the sustainable, high-quality software and services as is necessary for industry purposes. OSS poses new challenges that many companies can hardly bear. For instance, as mentioned earlier, OSS is following the agile mission that “less is more.” Instead of complex features as in most commercial software, it has only the bare minimum to improve quality and turn-around time. Gresse von Wangenheim and her colleagues show that using OSS in your products or using OSS-based tools means having the capability to enhance this basic functionality. Such an implementation is often complicated because development frameworks and documentation are insufficient.

OSS is certainly not free, and it comes with hidden features. Increasingly, once-independent open networks, including SourceForge.net, are part of industry alliances that decide what’s in and what’s out. Although alliances are a normal market mechanism, unclear governance processes and too many hidden players and rules will certainly reduce OSS’s attractiveness. Using components from these commercial OSS networks might create the very risks and dependencies we thought OSS would help to reduce. Dirk Riehle and his colleagues show one way of mitigation for bigger companies, namely setting up corporate communities that contribute to a variety of products and research themes. Smaller companies might be able to build their own community source models by including their customers and suppliers.

Today, OSS and its underlying principles are a major driver of industrial software engineering. Because of cost, quality, and efficiency reasons, software engineers and their managers need to understand these mechanisms, value chains, and business opportunities and become proficient regarding the advantages and constraints resulting from using OSS—as components, products, or tools. For a guided introduction to OSS with references to key articles, take a look at the IEEE TechSets on open source software. These online packages provide a practical introduction to OSS concepts together with useful guidance on applying OSS to your own development projects.

References