

Singapore Management University Institutional Knowledge at Singapore Management University

Research Collection School Of Information Systems

School of Information Systems

5-2005

Leveraging global resources: A distributed process maturity framework for software product development

Narayan RAMASUBBU

Singapore Management University, nramasub@smu.edu.sg


M. S. Krishnan

University of Michigan

Prasad Kompalli

DOI: <https://doi.org/10.1109/ms.2005.69>

Follow this and additional works at: https://ink.library.smu.edu.sg/sis_research

 Part of the [Computer Sciences Commons](#), and the [Management Information Systems Commons](#)

Citation

RAMASUBBU, Narayan; Krishnan, M. S.; and Kompalli, Prasad. Leveraging global resources: A distributed process maturity framework for software product development. (2005). *IEEE Software*. 22, (3), 80-86. Research Collection School Of Information Systems.

Available at: https://ink.library.smu.edu.sg/sis_research/120

This Journal Article is brought to you for free and open access by the School of Information Systems at Institutional Knowledge at Singapore Management University. It has been accepted for inclusion in Research Collection School Of Information Systems by an authorized administrator of Institutional Knowledge at Singapore Management University. For more information, please email libIR@smu.edu.sg.

Leveraging Global Resources: A Process Maturity Framework for Managing Distributed Development

Narayan Ramasubbu and M.S. Krishnan, *University of Michigan*

Prasad Kompalli, *SAP AG*

This evolutionary framework proposes 24 new key process areas essential for managing distributed software product development and continuously improving product management capabilities.

Leveraging global resources by distributing software development is becoming pervasive in the software industry. Chief among the motives are capitalizing on differences in labor cost, having a strategic regional presence for improved customer service, and building a diverse knowledge base. While the literature on distributed software development is still emerging, the use of geographically dispersed teams continues to grow, outpacing our understanding of their dynamics and the appropriate development processes.

Popular and well-tested software process frameworks, such as the Software Engineering Institute's Capability Maturity Model and ISO 9001, detail *key process areas* for software development. Each KPA is a small component of software development and includes a cluster of related activities to be performed collectively. Focusing more on continuous improvement in code development processes, generic-process frameworks such as the CMM lack KPAs that address capabilities for managing distributed software projects, such as establishing mutual knowledge and managing geographically dispersed social networks.

We drew on concepts from software engineering, collaboratory, and virtual-teams research to develop a practical process-maturity framework for managing distributed software development. We identify 24 new KPAs that

address the wide-ranging capabilities needed for managing such development and arrange them in an evolutionary order similar to the CMM framework. The evolutionary or phased approach in improving software management capabilities helps firms systematically assess their situations and plan for improvements. We also report the results of a statistically tested maturity assessment survey and test the overall rigor of our model against industry expert opinion and objective data collected from real-world projects implemented at SAP AG, a leading global-enterprise software firm.

Theoretical foundations and model development

The first step in building our framework was to draw key theoretical grounds from existing research. Work on geographically dispersed teams mainly emphasizes using infor-

mation technology to manage interdependent tasks across distance. Some researchers are extremely optimistic that modern information technology and communication systems will render geographical distance irrelevant to business,¹ but others caution that certain human and context-specific characteristics can never be replicated over distance—even with sophisticated technologies.² Gary and Judith Olson, while ascertaining that “distance matters,” synthesize four concepts that will be crucial for geographically distributed work: *collaboration readiness*, *common ground*, *coupling in work*, and *technology readiness*.²

Collaboration readiness, although originally referred to in the literature in terms of groupware technologies, has a broader meaning in our context. By collaboration readiness, we mean the ability of an overall software development governance model to set business goals that easily translate into maintainable interdependent tasks across geographically dispersed teams. Software industry projects range from limited-time maintenance service projects to reengineering projects, to dedicated product development projects. Varying software projects’ idiosyncratic characteristics require varying governance models and contractual schemes. A firm’s choice of a specific governance model at both the organizational and project levels lays the fundamental structure for distributed product development and plays an important role in influencing project performance. So, from collaboration readiness, we learn the need for distributed process-maturity KPAs that address forming and continuously improving appropriate governance models, contractual schemes, and business goals.

Common ground reflects the shared knowledge of distributed-development participants who reside in distant locations. Researchers have shown that developing such knowledge is a collaborative process.³ In a software development context, mutual knowledge could be about the software product’s history and legacy, company culture, common experience with customers, or even a basic understanding of the personnel’s working styles. Informal interactions in software development play a crucial role in disseminating information within a team. When personnel are separated by distance, establishing common ground is essential to avoid potential communication gaps that could deteriorate product development

performance. So, from the concept of common ground or mutual knowledge, we learn the need for KPAs for formal and social communication and knowledge management.

Coupling in work refers to the mechanisms for dividing labor in distributed product development. Modern software products are extremely complex systems; managing this complexity is a prerequisite for achieving efficient division of labor. Herbert Simon, in his classic article “The Architecture of Complexity,” details key aspects of complex systems that help you manage such systems.⁴ He argues that in nature, complexity frequently takes the form of hierarchy—a system composed of subsystems that in turn have their own subsystems—and that hierarchic systems evolve far more quickly than nonhierarchic systems of comparable size. Simon’s theory in a software context emphasizes the importance of the roles played by the developed product’s modularity and the development team’s modularity. The distributed product development process must result in guidelines for the software manager to dynamically control product modularity and the modularity’s span of remote teams. So, coupling in work highlights the importance of distributed process-maturity KPAs to address labor-division mechanisms.

Technology readiness refers to the development infrastructure and personnel capability levels for using collaborative technologies such as groupware communication systems, videoconferencing, synchronous multiauthoring tools, and group debugging tools. Often, the availability of such tools is mistaken for readiness and efficient use. As Wanda Orlikowski points out, efficient use of collaborative technologies depends on the organization’s structural properties (such as policies and reward structures) and the personnel’s understanding of technology and their work.⁵ Clearly, these two factors are also critical in determining the sustenance of successful distributed product development. So, technology readiness reveals the need for distributed process-maturity KPAs that emphasize monitoring the use of a technology infrastructure and its continuous improvement.

Our new KPAs build on these four theoretical concepts and our experience with distributed software development. Figure 1 shows our evolutionary, distributed process-maturity model in which we’ve mapped the KPAs to the four concepts.

When personnel are dispersed, establishing common ground is essential to avoid communication gaps that could deteriorate product development.

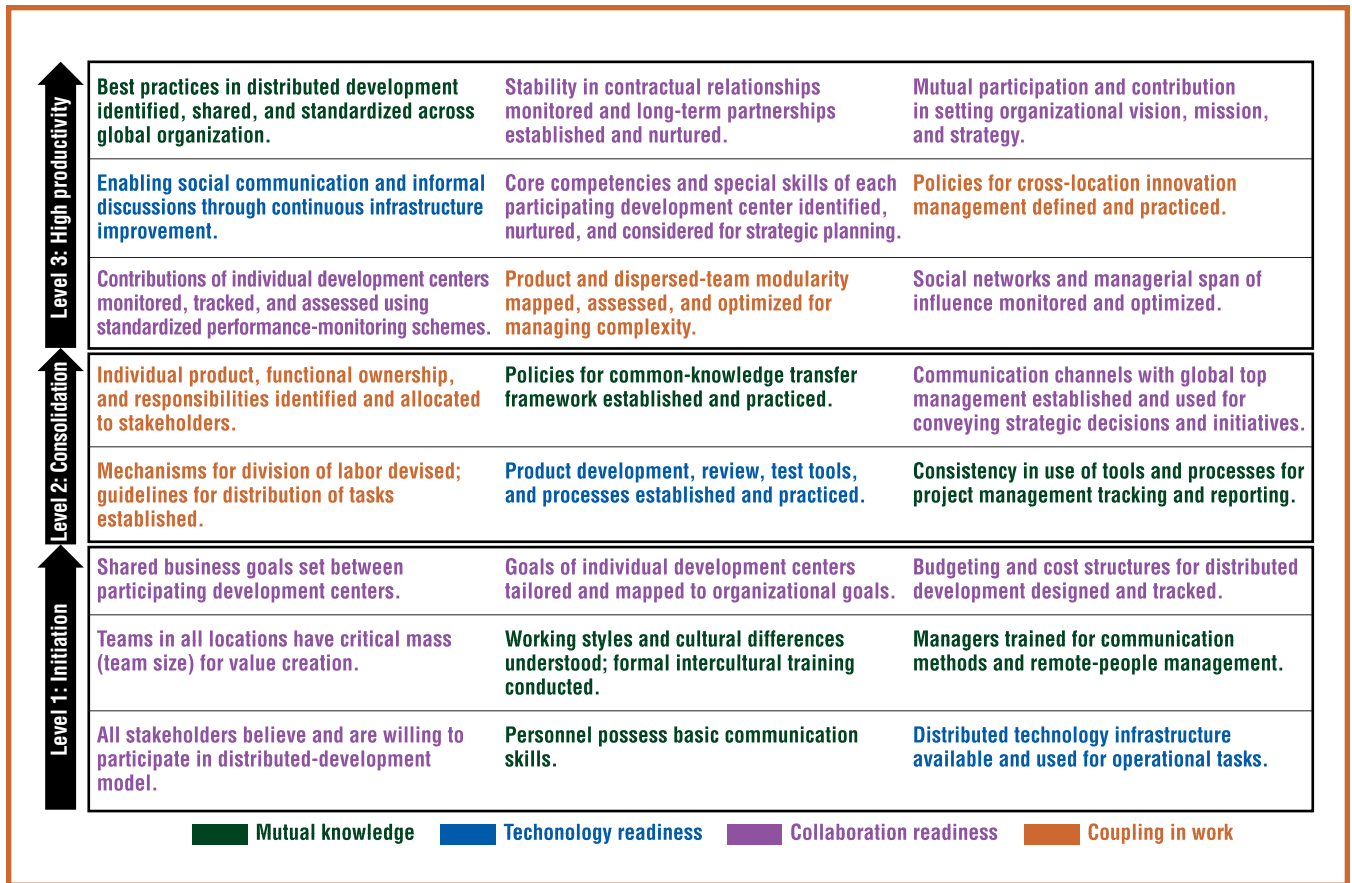


Figure 1. Our distributed process-maturity model features 24 new key process areas mapped to four theoretical concepts for distributed work: mutual knowledge, technology readiness, collaboration readiness, and coupling in work.

Operationalizing the model for practice

We implemented our distributed-development model in selected SAP global development teams. SAP, headquartered in Walldorf, Germany, is the world’s third-largest software supplier, employing over 28,900 people in more than 54 countries. Part of the packaged enterprise software industry, SAP develops software product solutions for 23 industrial sectors and has about 60,100 installations with 12 million end users. Over the past decade, SAP has increasingly adopted the distributed product-development paradigm to achieve its goals of establishing a strategic regional presence around the globe and tapping the knowledge base in various countries.

Three business reasons drove development and use of a distributed process-maturity model. First, with investments in the billions of euros in distributed product development, an organization must evaluate such ventures’ performance in terms of not only monetary gain but also improved product quality, development productivity, and achieved strategic business

goals. Second, popular software maturity models such as CMM and ISO 9001 don’t address the key processes required to develop or evaluate distributed product development that can be readily adopted. Third, similar to Total Quality Management, an organization must identify the best practices employed in specific regions and products and adopt them across the board. Thus, we developed our model to provide top management with a reliable framework for assessing, monitoring, and improving management practices for globally distributed team performance.

Validating the model

Before using our model for day-to-day operations, we validated it through in-depth interviews and review sessions with an expert committee. The committee comprised 34 randomly selected, high-level management executives consisting of certified quality assurance managers, development managers, program directors, and vice presidents of software product development. Committee members were from five distinct enterprise product lines and

Table 1**Our key process areas ranked by an expert review committee and compared to our model's rankings**

KPAs in our model	Maturity phase in our model	Expert rankings for level 1 match	Expert rankings for level 2 match	Expert rankings for level 3 match	Correct (%)
Belief and willingness	1	26	8	—	76.47
Personnel communication skills	1	30	4	—	88.24
Utilization of distributed technology infrastructure	1	30	4	—	88.24
Critical mass (team size)	1	28	5	1	82.35
Understanding cultural differences	1	30	3	1	88.24
Managerial training	1	17	17	—	50.00*
Setting shared business goals	1	27	3	4	79.41
Tailoring business goals	1	27	3	4	79.41
Budgeting and cost structures	1	28	—	6	82.35
Devising mechanisms for division of labor	2	28	6	—	82.35
Product development tools and processes	2	—	26	8	76.47
Consistency in project management processes	2	—	33	1	97.06
Ownership and responsibilities	2	—	30	4	88.24
Knowledge transfer	2	2	30	2	80.24
Top-management communication channels	2	13	6	15	17.65*
Performance monitoring	3	1	7	26	76.47
Managing complexity	3	1	3	30	88.24
Managing social networks	3	—	3	31	91.18
Enabling social communication via technology	3	—	5	29	85.29
Nurturing and leveraging core competencies	3	—	9	25	73.53*
Interorganizational innovation management	3	—	—	34	100.00
Best practices management	3	—	5	29	85.29
Contract management and nurturing partnership	3	—	4	30	88.24
Managing symbiotic relationship and continuous development	3	—	5	29	85.29

Average model acceptance = 80.43%

*Below cutoff value of 75% of expert acceptance

had an average of 12 years' industry experience. Participants in our review committee were in charge of distributed development projects that had customers in Germany, India, the US, Singapore, Indonesia, and Thailand.

In our review session, we first presented our project's background and objectives. We then provided our list of distributed-development KPAs randomly sorted and asked the participants to rank and order them along a maturity

path that consisted of our three evolutionary levels. We then compared our model with the experts' opinions (see Table 1).

Similar to previous research in this area,⁶ for validating our model, we used a cutoff of 75 percent as an acceptable matching level. As Table 1 indicates, our maturity model matched well with the experts'. However, three KPAs—managerial training, top management communication channels, and nurturing and leveraging

Table 2**KPA audit anchor definitions**

Response	Anchor definition	Response score
Almost always	The KPA practice is well established and consistently performed as a standard operating procedure (90–100% of the time).	5
Frequently	The KPA practice is performed relatively often but sometimes is omitted under difficult circumstances (60–90% of the time).	4
About half	The KPA practice is performed only about half the time (40–60% of the time).	3
Occasionally	The KPA practice is performed less often than not (10–40% of the time).	2
Rarely if ever	The KPA practice is rarely performed.	1
Does not apply	You have the required knowledge about the project and question but feel that this question does not apply.	—
Don't know	You are uncertain about answering this question and do not have the required information.	—

core competencies—didn't meet the acceptable cutoff level of 75 percent. We discussed these with the review committee.

The committee was divided on placing managerial training in the initiation or the consolidation phase. The primary point of debate was on the effectiveness of learning through formal training. Experts who placed managerial training in the initiation phase argued that learning before doing is critical to handle and lead virtual teams effectively and emphasized training as an absolute necessity before initiating distributed development projects. However, experts who placed managerial training in the consolidation phase argued that external or laboratory training can't be effectively tailored to meet contextual demands and emphasized that learning happens on the job. This group argued that the training KPA should be placed in the consolidation phase to facilitate dissemination of gathered knowledge. While learning by doing might be effective in certain scenarios, several studies have emphasized the benefits of defect-prevention activities such as process training that stress learning before doing.⁷ Also, managerial training to handle remote personnel has fundamental implications in terms of influencing employee motivation and morale. Thus, we left managerial training in the initiation phase.

The top-management communication channel KPA received almost equal acceptance for all the evolution levels. From this result, we inferred that this KPA needs finer analysis levels. That is, different modes of top-management participation must be analyzed separately to identify incremental process-maturity stratification. So, we decided to work on this for future versions of the model and dropped this KPA from our current analysis.

Nurturing and leveraging core competencies fell short of our acceptance cutoff level. Experts who placed this KPA in the consolidation phase argued that high productivity levels can't be achieved without a business model based on core competencies; hence, processes necessary for nurturing core competencies must be placed in the consolidation phase. However, core competencies can't emerge without sufficient ownership and responsibility structures in place. Furthermore, the consolidation phase is a transforming phase in which stable mechanisms for the division of labor emerge through interactions and a natural choice of tasks. In an intermediate stage, the fuzziness of "joint ownership" starts translating to individual responsibilities and goals that shape the development of long-term team structures. Only when a stable, long-term organizational structure is in place is it beneficial to identify the core competencies of the organization's individual components. The expert committee agreed with this viewpoint, and we decided to place the core competency KPA in the high-productivity phase. Future studies might verify this assertion.

Assessing distributed process-maturity levels

Once the experts validated the KPAs' positioning in individual maturity levels, we devised an assessment questionnaire for use in process maturity audits for distributed development projects. Similar to the CMM, each KPA had several goal questions to assess KPA implementation in the audited team. Each KPA audit question had seven possible responses: almost always, frequently, about half, occasionally, rarely if ever, does not apply, and don't know. These responses were scored on a frequency-based scale (see Table 2).

We, along with a certified quality assurance manager, audited a convenient sample of seven distributed-development teams. The teams develop large enterprise-software solutions in the domains of human capital management, customer relationship management, and the oil and gas industry. The teams' sizes and ages varied considerably. The sample included four large teams that employed more than 40 developers and three small teams that employed fewer than 15 developers. Two teams had been involved in distributed development for more than six years, three teams for four years, and two teams for three years. Although the overall length of practice of distributed development varied among teams, the distributed process-maturity framework in our study was initiated in the teams at the same time in August 2002. We obtained multiple responses for the audit survey from several personnel in the same project (typically a senior developer, project coordinator, and development manager) through structured interviews and surveys. We then assessed reliability of the KPA constructs and goal questions in these multiple responses. The minimum reliability among multiple responses was 0.8 as measured by Cronbach's Alpha. This reliability value is above the recommended threshold of 0.7.

Among the seven teams we audited, two teams were at the high-productivity level, two were at the consolidation level, and three were at the initiation level. We based these assessments on the extent to which the teams defined, measured, practiced, and managed KPAs in our framework. We couldn't make any rigorous conclusions on the association between team size and maturity level or on the relation between number of years the team has existed and maturity level. To validate the maturity-level assessments with objective data, we collected individual project metrics of quality and productivity from these teams. We measured quality using average defect density in the team (number of defects per KLOC) and measured productivity as the average processing time for an adjusted function point count (size of code from an end user's perspective adjusted for complexity). We then mapped these objective metrics with the teams' corresponding distributed process-maturity levels (see Figure 2).

Figure 2 shows that teams assessed at the high-productivity level according to our framework are in the sweet quadrant of high devel-

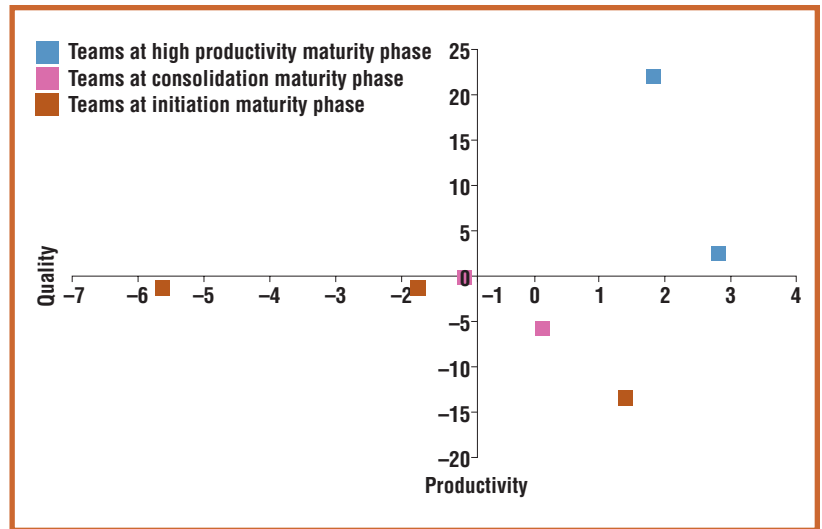


Figure 2. Mapping assessment results with project metrics.

opment productivity and high quality. These teams operate with well-defined distributed-development processes and deliver superior results. Teams assessed at consolidation phase maturity lie near the sweet quadrant with average levels of quality and productivity. Teams at initiation phase maturity show skewed results with poor quality or poor productivity. This mapping of project metrics data with the results of our assessment validates our evolutionary maturity framework.

As software firms gain capability in adopting and rigorously practicing disciplined development processes, it's also important to focus on continuous improvement in the management of software development projects. Such a focus will enable firms to adopt newer business models, such as distributed development, as market conditions demand.

As globally dispersed teams become pervasive, top management needs a framework to assess its performance and to initiate activities for continuous improvement in management of such teams. Our distributed process-maturity framework is a first attempt to satisfy this need and develop a rigorous distributed product development capability model.

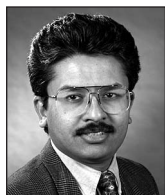
Future researchers could expand our framework to accommodate subtle features of distributed technology infrastructure and other management practices relevant to remote-teams management. Studying distributed-development cases from multiple organizations and other

About the Authors




Narayan Ramasubbu is a doctoral candidate at the Stephen M. Ross School of Business, University of Michigan, Ann Arbor. His research work focuses on software engineering economics, software development processes, software product architecture, design, standardization, and customization. He received his undergraduate degree in electronics and communication engineering from the Coimbatore Institute of Technology, India. Before joining the doctoral program at Michigan, he was a senior developer consultant at SAP AG, Walldorf, Germany. Contact him at D0263, Doctoral Studies Office, Davidson Bldg., Stephen M. Ross School of Business, Univ. of Michigan, 701 Tappan St., Ann Arbor, MI 48109; nramasub@umich.edu.

Prasad Kompalli is a technology adviser assistant to the Corporate Officer of the extended board management at SAP AG, Walldorf. His research interests involve enterprise-application product development, managing distributed processes, and innovations in globally distributed work environments. He received his BS in computer science and engineering from the Regional Engineering College, Warangal, India. Contact him at SAP AG, Neurotstrasse 16, 69190 Walldorf, Germany; p.kompalli@sap.com.



M.S. Krishnan is a Michael R. and Mary K. Hallman E-Business Fellow, an area chair, and a professor of business information technology at the Stephen M. Ross School of Business, University of Michigan, Ann Arbor. His research interests include corporate IT strategy, IT's business value, software engineering economics, metrics and measures for productivity, quality, and customer satisfaction. He received his PhD in information systems from Carnegie Mellon University. Contact him at the Business Information Technology Dept., Stephen M. Ross School of Business, Univ. of Michigan, 701 Tappan St., Ann Arbor, MI 48109-1234; mskrish@bus.umich.edu.

software development contexts (such as services and system products) might identify drawbacks and shortcomings of our simple framework and pave the way for more comprehensive process schemes. 

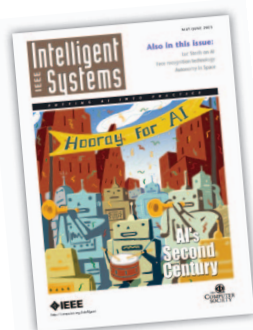
Acknowledgments

Several managers and developers at SAP Labs India and SAP AG provided necessary data and support for this research. Martin Prinz, Andre Wachholz-Prill, Alain Lesaffre, and Judy Olson provided valuable comments on improving our research design. A Michael R. and Mary Kay Hallman Fellowship at the University of Michigan Business School provided partial financial support for this research.

References

1. F. Cairncross, *The Death of Distance: How the Communications Revolution Will Change Our Lives*, Harvard Business School Press, 1997.
2. G.M. Olson and J.S. Olson, "Distance Matters," *Human Computer Interaction*, vol. 15, 2000, pp. 139-179.
3. H.H. Clark and S.E. Brennan, "Grounding in Communication," *Perspectives on Socially Shared Cognition*, L. Resnick, J.M. Levine, and S.D. Teasley, eds., Am. Psychological Assoc., 1991, pp. 127-149.
4. H.A. Simon, "The Architecture of Complexity," *Proc. Am. Philosophical Soc.*, vol. 106, no. 6, 1962, pp. 467-482.
5. W. Orlikowski, "Learning from Notes: Organizational Issues in Groupware Implementation," *Proc. ACM Conf. Customer-Supported Cooperative Work (CSCW 92)*, ACM Press, 1992, pp. 362-369.
6. M.S. Krishnan and M.I. Kellner, "Measuring Process Consistency: Implications for Reducing Software Defects," *IEEE Trans. Software Eng.*, vol. 25, no. 6, 1999, pp. 800-815.
7. P. Pisano, "Knowledge, Integration, and Locus of Learning: An Empirical Analysis of Process Development," *Strategic Management J.*, vol. 15, 1994, pp. 85-100.

See the Future of Computing Now in *IEEE Intelligent Systems*



Tomorrow's PCs, handhelds, and Internet will use technology that exploits current research in artificial intelligence. Breakthroughs in areas such as intelligent agents, the Semantic Web, data mining, and natural language processing will revolutionize your work and leisure activities. Read about this research as it happens in *IEEE Intelligent Systems*.



SUBSCRIBE NOW! <http://computer.org/intelligent>

IEEE Intelligent Systems