

# **RISE – Reuse im Software Engineering**

Ralph Traphöner  
empolis GmbH  
Europaallee 10  
D-67657 Kaiserslautern  
ralph.traphoener@empolis.com

## **Kurzfassung**

Bitte an dieser Stelle eine Kurzfassung setzen, die den wesentlichen Inhalt des Beitrages widerspiegelt und eine Beurteilung und Bewertung der erreichten Forschungsergebnisse ermöglicht. Dieser Teil sollte eine halbe Seite nicht überschreiten.

## **1. Einleitung**

Dynamische Forschungs- und Entwicklungsgebiete wie das Software Engineering sind in starkem Maße auf die Wiederverwendung von Produkten, Methoden, Modellen und Erfahrungen angewiesen. Dies dient nicht nur der Unterstützung der Softwareentwicklung mit ingenieurmäßigen Mitteln sondern auch der Weiterentwicklung ihrer Techniken, Methoden und Werkzeuge.

Seit den 80er Jahre etablierte sich im Bereich Software Engineering zunehmend der "Experience Factory" (EF) Ansatz unter der Federführung von Victor Basili [4][3], der explizit das kontinuierliche (organisatorische) Lernen aus Erfahrungswissen zum Gegenstand hat. Charakteristisch für eine EF ist die Trennung von Lern- und Projektorganisation, da sich in der Praxis gezeigt hat, dass dies für die Unterstützung organisationalen Lernens wesentlich ist [4]. Der EF zugrunde liegt das Quality Improvement Paradigma, ein zielorientierter Lernzyklus zur erfahrungsbasierten Verbesserung von Projektplanung, Projektdurchführung und Projektlernen. Zielorientiertes Messen und Bewerten ist dabei die systematische Vorgehensweise zur Evaluation [5][7]. Mittlerweile ist im Bereich Experience Factory durch eine "tiefe", d.h. die kognitionswissenschaftlichen Grundlagen ausnutzende Integration mit dem Case-Based Reasoning Ansatz ein erster wichtiger Fortschritt erzielt worden [15][2].

Das initiale frühe Beispiel für eine operationale EF ist das NASA Software Engineering Laboratory (SEL) [6]. Mittlerweile gibt es mehrere EF-Anwendungen sowohl in den USA als auch in Europa [9][11][13][10][16]. Eine Integration dieses Konzeptes in deutschen KMUs blieb allerdings bislang aus.

Das Organisationsgedächtnis wird technisch durch ein Organisational Memory Information System (OMIS, kurz OM) unterstützt. Wir verstehen darunter ein Computersystem, das in der Organisation Wissen und Informationen fortlaufend sammelt, aktualisiert, strukturiert und für verschiedene Aufgaben möglichst kontextabhängig, gezielt und aktiv zur Verbesserung des kooperativen Arbeitens zur Verfügung stellt. Obwohl die Organisationslehre schon seit einigen Jahren die Basisfunktionalitäten des OM diskutiert hat [14], existieren umfassende soft-

waretechnische Konzepte und Analysen von deren Einbettung in den organisatorischen und individuellen Arbeitskontext erst seit kurzem. Deutschland - und insbesondere auch die Antragsteller - spielen dabei eine Vorreiterrolle sowohl in der theoretischen Durchdringung [12][1], als auch in der Verwendung innovativer Technologien für kontextorientierte und prozessintegrierte Informationserfassung und -nutzung, verteilte Systemdienste und selbst organisierende und adaptive Funktionen [8]. Erste Ergebnisse dieser Forschungen sind im Prototypstadium auf dem Sprung in die Anwendung oder in Produktfunktionen, es fehlen jedoch noch fundierte Erfahrungen über Einführungsprozesse und Akzeptanz, genauso wie breit angelegte, lang laufende empirische Studien über die Auswirkungen solcher Systeme auf Arbeitsqualität und -effizienz. Ferner sind OMIS bis dato ausschließlich als domänenunabhängige Systeme gestaltet worden. Es ist offen, inwieweit man durch das Zuschneiden auf das Feld des Software Engineering positive Effekte für Umsetzbarkeit, Wiederverwendbarkeit und Effizienz von Lösungen erzielen kann.

Bisherige Arbeiten in diesem Bereich, welche den Stand der Praxis näher an den Stand der Wissenschaft führen, sind:

- indiGo: „Integratives Software Engineering durch diskursunterstützende Groupware“ verbindet moderierte Diskussionsgruppen zur Einführung und Inspektion von Geschäftsprozessen zur Unterstützung prozessbezogenen Lernens innerhalb einer Organisation
- CoIN: „Corporate Information Network“ stellt Methoden und Technologien für die Wiederverwendung von Erfahrungen zu Geschäftsprozessen bereit.
- Inreca II: „Information and Knowledge Re-engineering for Reasoning from Cases“ stellt Vorgehensweisen für Anwendungen im Bereich fallbasierte Entscheidungsunterstützung zur Verfügung.
- Softquali: „Systematische Softwarequalitätsverbesserung“ durch zielorientiertes Messen und Bewerten sowie explizite Wiederverwendung von Erfahrungswissen stellt Wiederverwendungsszenarien für SE-Erfahrungswissen bereit.

## 2. Projektziele und Status

Die Entwicklung von Software ist eine hochgradig wissensintensive Tätigkeit bei der viele Produkte, Methoden und Modelle erstellt sowie Erfahrungen gesammelt werden. Die Produktivität als auch die Qualität der Ergebnisse hängen stark davon ab, in welchem Maße vorhandenes Wissen aus anderen Projekten und dem wissenschaftlichen Bereich Software Engineering zur Verfügung steht und tatsächlich angewendet wird. Die schnelle Verfügbarkeit von relevantem Wissen steigert die Produktivität des Unternehmens und verhindert das Entstehen von Wissens- und Kompetenzlücken, üblicherweise verursacht durch:

- Wissensschwund: Entwickler verlassen das Unternehmen und nehmen Erfahrungen und Expertise über Produkte des Unternehmens mit.
- Wissensreplikation: Lösung aus Projekt X wird in Projekt Y – obwohl möglich – nicht wieder verwendet, sondern vollständig neu entwickelt. Defekte und Lösungen aus Projekt A werden in Projekt B wiederholt.
- Innovationsrückstand: Konkurrenten im In- und Ausland verwenden neuere (und bessere) Technologien, an denen Kunden sich orientieren

Erfolgskritisches wiederverwendbares Wissen im Software Engineering ist von ganz unterschiedlichem Charakter. Meist ist es implizit in den Köpfen und Gewohnheiten der Mitarbei-

ter, selten aber auch explizit, in unterschiedlichen Formen vorhanden, wie z. B. Methoden, Produkten, Defekten oder Technologiewissen.

Diese Wissensseinheiten sind deshalb besonders wertvoll, weil sie Erfahrungswissen verkörpern, welches Software-Ingenieure in einer Vielzahl von Projekten gewonnen haben, und das so verallgemeinert wurde, dass es auch in neuen Projekten anwendbar ist.

Eine informationstechnische Unterstützung für die Wiederverwendung der Produkte, Methoden, Modelle und Erfahrungen bei der Softwareentwicklung hat die Aufgabe, relevantes Wissen zu sammeln, aufzubereiten und den jeweiligen Software-Ingenieuren zur Verfügung zu stellen. Ein dafür geeignetes System muss neben der Bereitstellung der didaktisch aufbereiteten Wissensinhalten insbesondere folgende Punkte berücksichtigen:

- Inhalt: Welches Wissen erheben, aufbereiten und weitergeben?
- Methode: Wie Wissen erheben, aufbereiten, weitergeben und evaluieren?
- Rolle: Wer erhebt / bereitet auf / gibt weiter / evaluiert / erhält Wissen?
- Fluss: Woher (Von wem) Wissen erheben und an wen weitergeben?
- Zeitpunkt: Wann Wissen erheben, aufbereiten, weitergeben oder evaluieren?

Der Ausgangspunkt für die zu entwickelnde Methodology und Technology ist eine Fusion von zwei Modellen, die am IESE bzw. am DFKI entwickelt wurden.

Ziel von RISE ist es, eine umfassende Lösung für die Wiederverwendung von Produkten, Methoden, Modellen und Erfahrungen im Software Engineering zur Verfügung zu stellen. Ausgangspunkt für diese Entwicklungsarbeit bildet der Umgang mit Defekten und Feature (interne Aufträge) in der Softwareentwicklung. Dadurch bildet sich eine speziell auf KMUs zugeschnittene Herangehensweise für ein praktikables Software Engineering. Diese Herangehensweise kann darüber hinaus Ausgangspunkt für weitere Software-Engineering Aktivitäten sein.

Der Begriff "Auftrag" bezeichnet dabei eine Bandbreite von Entwicklungstätigkeiten, beginnend bei einfachen Debugging Aufgaben im Umfang weniger Stunden, bis hin zu Entwicklungen von Komponenten im Umfang von 30 und mehr Tagen.

Im Gegensatz zu den bisherigen Experience Factory Implementierungen wird in RISE ein verstärktes Augenmerk auf agile Methoden und kleine Softwareentwicklungsteams gelegt. Das Ziel dabei ist, den Mehraufwand für die zu entwickelnden Methoden zu minimieren und zu einer möglichst transparenten Integration in den Entwickleralltag und die Teamarbeit zu gelangen.

Der technische Ansatz der hierzu gewählt wurde (siehe Abbildung) stellt ein Wiki in das Zentrum, das um eine assoziative Suche im Internet und eine fallbasierte Suche in den lokalen Inhalten ergänzt wird. Die RISE Plattform wird diese Technologien nutzen und ergänzen, um zum einen die Erschließung des Internet als externe Wissensquelle zu systematisieren und zum anderen die iterative Formalisierung des SE Wissens, das im Wiki gesammelt wird, zu ermöglichen. Dieses formalisierte Wissen wird genutzt, um den Informationszugriff im Wiki zu verbessern und dadurch die Produktivität der Entwickler zu steigern.

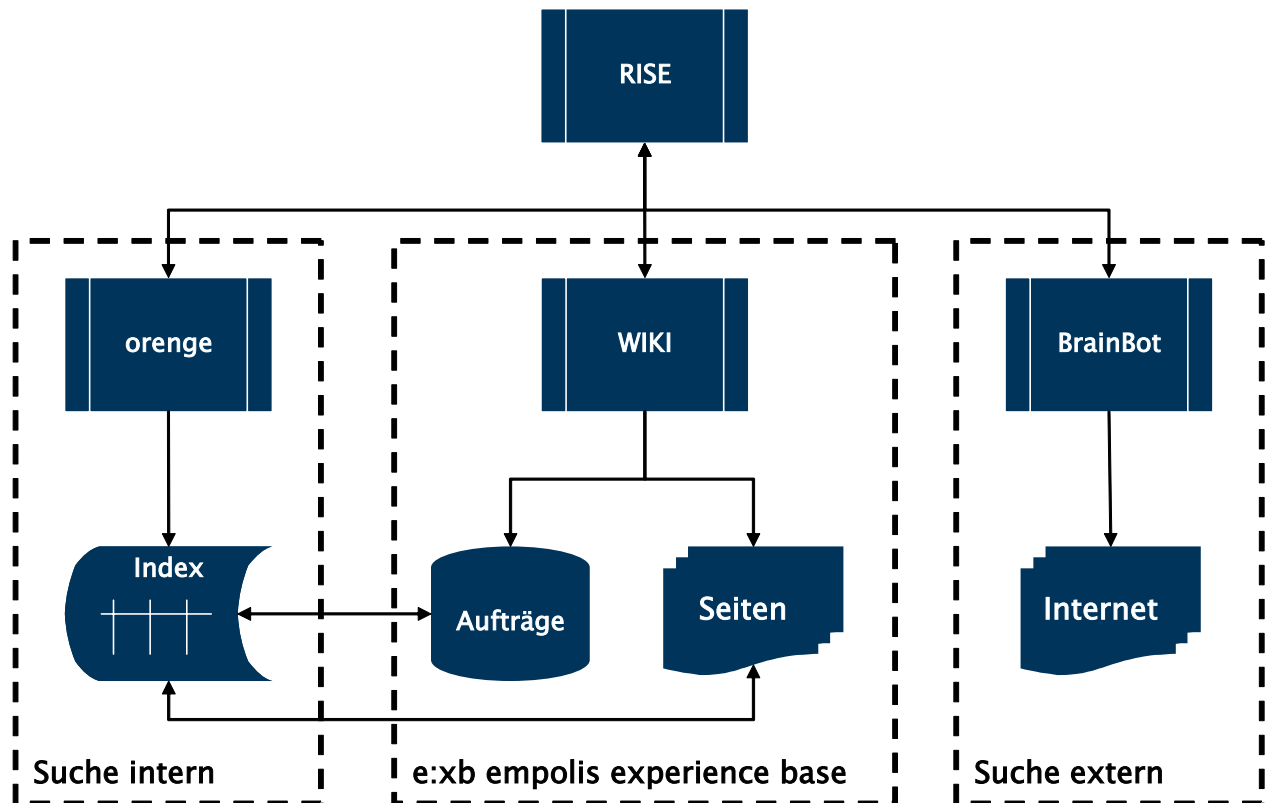


Abbildung: Architektorentwurf

Das Wiki selbst stellt in diesem Kontext die zentrale Kollaborations Plattform der Entwickler dar. Hier werden Aufträge inhaltlich und administrativ bearbeitet. Hier werden alle wichtigen Inhalte zu Technologien, Produkten, Kundenaufträgen und Problemlösungen erstellt und verfügbar gemacht.

### 3. Erste Erfahrungen und Ausblick

Im Februar wurde bei der empolis, als dem Hauptanwendungspartner, ein Wiki in Betrieb genommen, das im weiteren Verlauf des Projekts um die anderen Komponenten und Funktionen erweitert wird. Mit diesem System arbeiten regelmäßig 60 Entwickler. Bis heute - Stand Juni 2004 - wurden mehr als 5000 Seiten in das System eingestellt. Die Akzeptanz des Wiki Paradigmas war für alle Beteiligten überraschend hoch.

Aufbauend auf den somit realen Inhalten und der Möglichkeit zur Beobachtung und Analyse der Entwicklertätigkeit mit dem System bietet sich ein ausgezeichnetes Potential die zu entwickelnden Methoden in einer realistischen Umgebung zu erproben und Nutzer gerecht auszugestalten.

Die Inhalte des Systems und die Arbeitsweise der Entwickler zeigen bereits jetzt, dass der gewählte Ansatz viel versprechend ist. Dabei ist entscheidend für den Erfolg, dass der gefühlte Mehraufwand weit unter dem gefühlten Nutzen der Anwender bleibt.

## Referenzen

- [1] Abecker, A., Hinkelmann, K., Maus, H., Müller, H.-J., Hrsg. (2002). "Geschäftsprozessorientiertes Wissensmanagement". Heidelberg: Springer Verlag, Serie xpert Press.
- [2] Althoff, K.-D. (2001). Case-Based Reasoning. In S. K. Chang (ed.), Handbook of Software Engineering and Knowledge Engineering, Vol. 1, World Scientific, pp. 549-588.
- [3] Althoff, K.-D., Birk, A., Hartkopf, S., Müller, W., Nick, M., Surmann, D. & Tautz, C. (2000). Systematic Population, Utilization, and Maintenance of a Repository for Comprehensive Reuse. In G. Ruhe & F. Bomarius (Eds.), Learning Software Organizations - Methodology and Applications, Springer Verlag, LNCS 1756, 25-50.
- [4] Basili, V.R., Caldiera, G. & Rombach, D. (1994). Experience Factory. In Marciniak, J.J. (ed.), Encyclopedia of Software Engineering, vol 1, 469-476. John Wiley & Sons.
- [5] Basili, V.R., Caldiera, G. & Rombach, H.D. (1994a). Goal-Question-Metric Paradigm. In Marciniak, J.J. (ed.), Encyclopedia of Software Engineering, vol 1, 528-532.
- [6] Basili, V.R., Caldiera, G., McGarry, F., Pajersky, R., Page, G., Waligora, S. (1992). The Software Engineering Laboratory - An Operational Software Experience Factory, 14th International Conference on Software Engineering, May 1992.
- [7] Briand, L.C., Differding, C.M. & Rombach, H.D. (1996). Practical guidelines for measurement based process improvement. Software Process 2(4), 253-280, Dec. 1996.
- [8] Dengel, A., Abecker, A., Bernardi, A., Elst, L. van, Maus, H., Schwarz, S., Sintek, M. (2002). "Konzepte zur Gestaltung von Unternehmensgedächtnissen", Künstliche Intelligenz, No. 1/2002, S. 5-11.
- [9] Haley, T.J. (1996). Software process improvement at Raytheon. IEEE Software 13(6), 33-41.
- [10] Houdek, F., Schneider, K. & Wieser, E. (1998). Establishing experience factories at Daimler-Benz: An experience report. Proc. 20th Internat. Conf. on Software Engineering (ICSE'98).
- [11] Humphrey, W.S., Snyder, T.R. & Willis, R.R. (1991). Software process improvement at Hughes Aircraft. IEEE Software 8, 11-23.
- [12] Lehner, F. (2000). "Organisational Memory - Konzepte und Systeme für das organisatorische Lernen und das Wissensmanagement". C. Hanser Verlag, München.
- [13] Seshagiri, G. (1996). Continuous process improvement: Why wait till level 5? Proc. 29th Hawaii Internat. Conf. on System Sciences, 681-692, IEEE Computer Society Press.
- [14] Stein, E. W., Zwass, V. (1995). "Actualizing Organizational Memory with Information Systems". Information Systems Research, 6(2), S. 85-117.
- [15] Tautz, C. (2000). Customizing Software Engineering Experience Management Systems to Organizational Needs. Doctoral Dissertation, Department of Computer Science, University of Kaiserslautern, Germany.
- [16] Tautz, C. & Althoff, K.-D. (2000). A Case Study on Engineering Ontologies and Related Processes for Sharing Software Engineering Experience. Proceedings 12th International Conference on Software Engineering and Knowledge Engineering (SEKE'00).
- [17] Argyris, C.; Schön, D.A. (1999). Die Lernende Organisation. Stuttgart: Klett-Cotta.
- [18] Arnold, R. (2000). Das Santiago-Prinzip: Führung und Personalentwicklung im lernenden Unternehmen. Köln: Dt. Wirtschaftsdienst.

- [19] Dreyfus, H.L.; Dreyfus, S.E. (1986). *Mind over machine: the power of human intuition and expertise in the era of the computer*. New York: Free Press.
- [20] Spada, H.; Mandl, H. (1988). *Wissenspsychologie: Einführung*. In: Mandl., H.; Spada, H. (Hrsg.): *Wissenspsychologie*. München, Weinheim: Psychologie-Verl.-Union, S. 1-18.