Effiziente und hochqualitative Softwareentwicklung mit

SPARX SYSTEMS

ENTERPRISE ARCHITECT

dem Modellierungswerkzeug, das den *kompletten* Projektzyklus abdeckt.

Professionelle Softwareentwicklung ist ein arbeitsteiliger Prozess an den hohe Qualitätsanforderungen gestellt werden. Anforderungserhebung, Anforderungsmanagement, Systemarchitekturplanung, Systementwurf, Implementation, Testen und Ausführungsplanung unterliegen immer strengerer Standardisierung. Kontinuierlich aktuell gehaltene Dokumentation, zeitparalleles Arbeiten im Team, Qualitätskontrolle, Nachvollziehbarkeit der Bedeckung von Anforderungen, Abschätzung und Handhabung von Risiken, Ressourcenplanung, Versionierung und transparentes Projektmanagement sind notwendige Anforderungen. Diese Funktionen und die logischen Vernetzungen – alle lösungs- und projektrelevanten Abhängigkeiten – in einem Werkzeug zu konzentrieren, beschleunigt und erleichtert die Projektumsetzung wesentlich. Enterprise Architect bietet diese Leistung zu günstigen Kosten – das gesamte Projektteam kann mit dem Werkzeug ausgestattet werden.

Requirements Management[1]**:=** Erfassung, Aufbereitung, Abstimmung, Verwaltung, Versionierung und Dokumentation von **Anforderungen** an das zu entwickelnde System und (nachvollziehende) Sicherstellung der Bedeckung und Einhaltung der Anforderungen während einer Systementwicklung.[2]

Welchen Nutzen bietet Anforderungsmanagement? Dient die Auflistung der Merkmale der geforderten Systemlösung nur der Qualitätssicherung im Sinne einer Absicherung des vereinbarten Lieferumfangs und der Prüfbarkeit der Erfüllung und damit auch als Abnahmekriterium? Ersetzen Anforderungslisten Ablaufbeschreibungen, UseCases oder gar andere Punkte der Anforderungsdokumentation? Ist Anforderungsmanagement abhängig vom Bereich und Gegenstand der Systementwicklung (Embedded Lösungen, kaufmännische Applikationen, usw.) unterschiedlich zu handhaben? Anforderungsmanagement ist elementarer Prozessbestandteil von System-Reifegrad-Modellen - wie ist es optimal in die Analysephase einzubinden oder betrifft es ausschließlich Design und Implementation? Welche Zusammenhänge bestehen zwischen Anforderungsmanagement, dem Vorgehen im Entwicklungsprozess und der (objektorientierten) Systemmodellierung? Welche Überlappungen treten auf, bedeuten sie Mehraufwand oder können sie vermieden werden? Wie kann die Lesbarkeit von Anforderungslisten sichergestellt werden, wie kann die Übersichtlichkeit gewahrt werden?

Was soll ein Werkzeug bieten, das Anforderungsmanagement unterstützt?

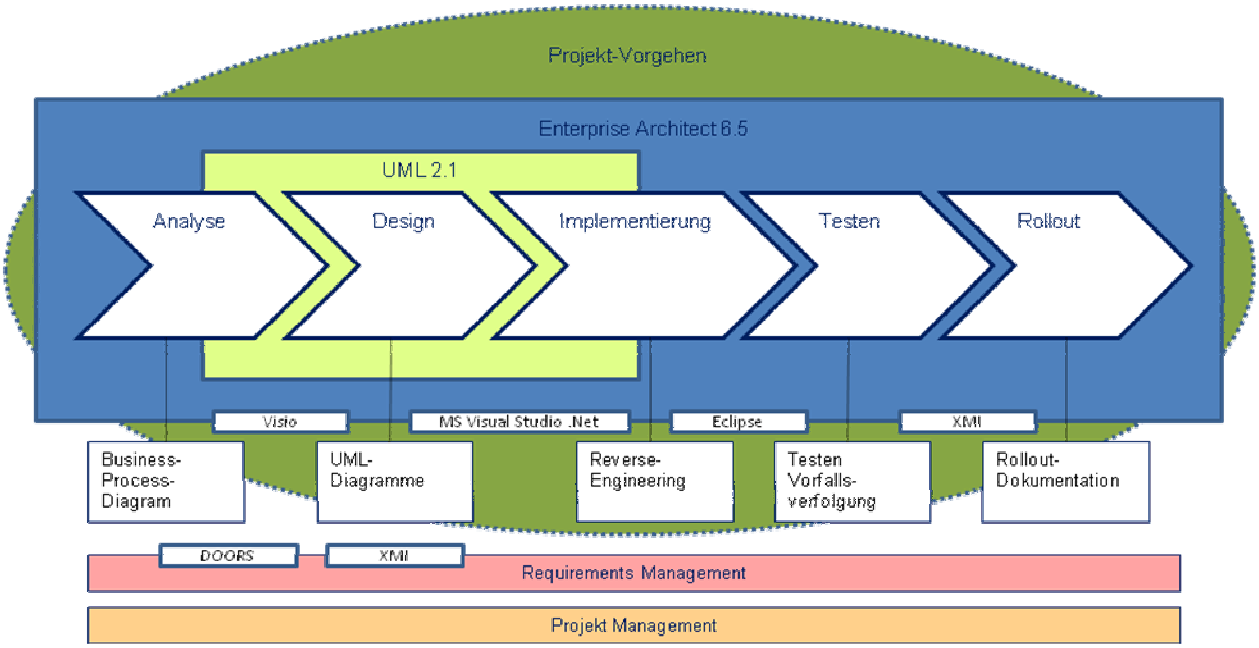
Qualitätssicherung ist notwendig. Prozessorientierung ist sinnvoll. Modellierung beschleunigt den Entwicklungsprozess und sichert Wiederverwendbarkeit. Jeder dieser Bereiche kann missbräuchlich gelebt werden und zum Selbstzweck entarten. Systeme und Projekte sind komplex. Vereinfachende Aufbereitung ist notwendig. Das Verständnis der Entscheidungsträger für die Inhalte und ihre Zeit, die sie den Unterlagen und dem Projekt widmen können, sind begrenzt.

---

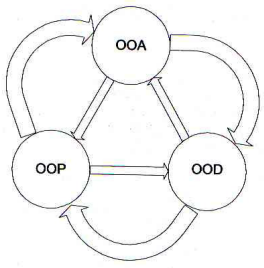
[1] „Requirements Management“ und „Anforderungsmanagement“ werden in gleicher Bedeutung verwendet

[2] Als elementare Prozesse in den Software- und System-Reifegrad-Modellen CMMI® und ISO/IEC 15504 (SPICE) sowie im Standard ISO/IEC 12207 enthalten.

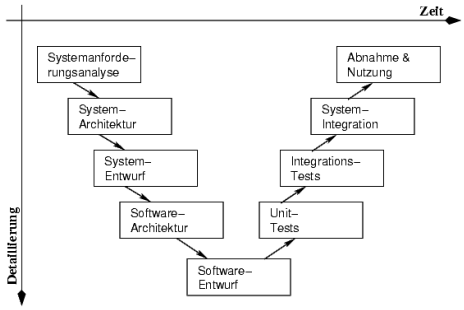
Sind Vereinfachung und Strukturierung möglich? Verschaffen wir uns einen Überblick!



Zunächst fällt auf, dass **Modellierungswerkzeuge** mit ihren Standards *Sprachen* und keine *Prozesse* darstellen (Abb. 1). UML 2.1 (www.uml.org) ist eine Sprache, sie definiert Symbole, Diagrammtypen für spezielle Zwecke - zur Visualisierung und Dokumentation, Strukturierung, Analyse, Design, usw.. Bestimmte Vorgehensweisen im Projekt zu wählen, z.B. anwenderzentriert mit der Erhebung von funktionalem Bedarf zu beginnen – und dabei auch Requirements zu identifizieren -, liegt nahe, ist aber keineswegs durch UML vorgegeben. Modellierungswerkzeuge werden also innerhalb eines Vorgehens, innerhalb eines von ihnen unabhängigen *Systementwicklungsprozesses* eingesetzt. Dieser *Prozessverlauf* kann reglementiert und auch zertifiziert sein oder aber dem Projektleiter überlassen sein.



Viele verschiedene **Vorgehensmodelle** sind verfügbar, werden zumeist heftig diskutiert und mehr oder minder erfolgreich gelebt. Der Marktdruck, eine Zertifizierung vorweisen zu können, motiviert zur Prozessorientierung. Sicher ist nur: Nicht jede Vorgehensweise passt zu jeder Aufgabenstellung. Rapid Prototyping oder iteratives Vorgehen (Abb. 2) werden wahrscheinlich schlecht zu einem „Hochhausbau" passen, Oberflächenoptimierung mit einem starren V-Modell (Abb. 3) zu betreiben, wird auch unbefriedigend sein. Ebenso ist offensichtlich, dass Prozessorientierung sehr oft sinnvolle Flexibilität verhindert, weil zumeist schlussendlich nur noch auf dem niedrigsten Detaillierungsgrad – überblickslos - umgesetzt wird.



Bei der Festlegung von **Prozessrichtlinien** für Systementwicklungsaufgaben ist also große Vorsicht geboten. Die Idee, mit einem einzigen, standardisierten Prozess – Anforderungsmanagement mit eingeschlossen - in einem Entwicklungsunternehmen für alle Projekte auszukommen, ist gefährlich. Auch die Annahme, Qualität schon ausschließlich durch die Einhaltung von Formalismen erzielen zu können, ist abstrus und falsch. In der Praxis ist sie durchaus öfters zu beobachten. Jedenfalls beeinflusst das gewählte Vorgehensmodell die Anforderungen an ein Anforderunsgmanagement-Werkzeug wesentlich.

Auch die ganz einfachen, **pragmatischen Randbedingungen**, denen alle Vorgehensmodelle unterliegen, haben starken Einfluss auf die Form und den Umfang von Anforderungsaufstellungen.

Welcher Festschreibungsgrad muss bei Angebotslegung, bei Auftragsannahme erreicht sein? In welcher Gliederung? Handelt es sich um eine lösungsorientierte Ausschreibung, d.h. liegt die Verantwortung für die Erreichung der gewünschten Ziele mittels der vorgegebenen Lösung beim Ausschreibenden oder muss auch der Auftragnehmer Verantwortung für die Wirkungserreichung übernehmen? Ist das Projekt in Analyse-/ Beratungs-, Design- und Umsetzungsphase mit getrennter Beauftragung, Abnahme und Abrechnung geteilt? Welche Kompetenzen hat der Auftraggeber? Wie sind diese verteilt? Wo liegt die Verantwortung für einzelne Anforderungspunkte? Wie werden Risiken adressiert, in Bezug gestellt, bewertet und verantwortet?

Umfassende Bedeckung zu fordern ist sinnvoll. Die Maximalanforderung entsteht, wenn der Umsetzer auch in die Aufgabendefinition mit eingebunden ist, bzw. – was häufiger auftritt – diese gemeinsam mit dem Auftraggeber aufzuarbeiten hat.

An dieser Stelle ist es von Vorteil, zu einer methodischen, **systemischen Ansicht** überzugehen – und nach Literatur, bzw. Standards zu fragen[3]:

1. Systemdesign ist methodisch betrachtet ein Entwurfsvorgang
2. Entwurfstätigkeit kann kategorisiert werden und
3. für die einzelnen Entwurfskategorien lassen sich Qualitätsmerkmale und Implikationen aufzählen

Natürlicher Entwurf ist gekennzeichnet durch die (spontane) Verwendung verfügbarer Materialien zur Deckung eines akuten Bedarfs. Wir denken sofort an primitive Kulturen. Halbe Kokosnüsse werden zu Tellern, Lehm und Stroh zu Baumaterial. Diese Vorgehensweise wird abgesehen von ökologischen Überlegungen in professionellem (System-)Design keine Rolle spielen. Dennoch kann man sie tagtäglich beobachten. Denken Sie an KollegInnen, die sich die Bildschirmhöhe ihres PCs durch Unterschieben von Kopierpapier millimetergenau justieren.

Ingenieurmäßiges Vorgehen ist gekennzeichnet durch die kombinatorische Verwendung – zumeist genormter – vorgefertigter Teile, Werkzeuge oder Methoden. Sei es bei der Konstruktion oder auch nur in deren produktiven Verwendung. Dies ist die Hauptschiene unserer Kultur, aber auch die Ursache für das unendlich vielgestaltige Flickwerk, das uns umgibt.

Denken Sie an einen Baumeister, Sie gehen zu ihm, um ihn nach einem zu bauenden Haus zu fragen. Er wird Ihnen seinen Musterkatalog zeigen, wird Ihnen zugestehen, im Detail noch Änderungen vorsehen zu können; Sie wählen aber schlussendlich aus Mustern aus…. Der Satz „Für unerwünschte oder schädliche Wirkungen fragen Sie bitte….." fällt nicht. Viele Wirkungen Ihrer Auswahlentscheidung werden Sie vorab nicht kennen, manche vielleicht nie bemerken. Manche werden angenehm sein, andere werden Sie hinnehmen.

Besonders gefährlich wird dieses Vorgehen, wenn nur einzelne Symptome befriedigt werden: Sie wollen ein Gerät zum Ausdrucken von Rechnungen mit 3 Durchschlägen. Sie fahren glücklich mit ihrem neuen Nadeldrucker vom Händler ins Büro. Kaum angeschlossen, lernen Sie, dass Nadeldrucker laut sind …. Später lernen Sie, dass Schallschutzhauben mühsam zu öffnen sind und dem Drucker thermische Probleme machen ….

Aus einem anderen Blickwinkel kann man diese Vorgehensweise auch unter dem Begriff „Denken in Lösungen" zusammenfassen. Wirklich Neues, mit hoher Wertschöpfung entsteht so kaum; immer wieder werden unbeabsichtigte Nebenwirkungen – zum Teil vorerst unerkannt - mit implementiert. Es entsteht Flickwerk, weil gewünschte Zusatzeigenschaften oder Optionen *bloß* (ein einzelnes Symptom behebend) nachgerüstet werden.

Architektonisches Vorgehen ist gekennzeichnet durch eine umfassende Analyse der gewünschten Zielwirkungen, Ergebnisse und funktionalen Eigenschaften (nicht der Funktionsweise!) – genau genommen unter temporärem Ausschluss des Andenkens von Lösungsansätzen. Ein guter Architekt

[3] Quelle: Univ. Prof. Heinz Zemanek , „Abstrakte Architektur", Vorlesung an der TU-Wien

analysiert Ihren (abstrakten) Bedarf. Nur dadurch, dass er sich auf die gewünschte Wirkung konzentriert, kann er Ihnen für Ihren Wohnbedarf – im Gegensatz zum Baumeister - zu einem großen Wohnwagen oder zum wohnlichen Ausbau eines Verkehrsflugzeugs raten – wenn er erkennt, dass dies Ihre Anforderungen besser erfüllt.

Kreativität selbst kann natürlich nicht schematisiert werden, aber die Voraussetzungen zu kreativen Einsichten können methodisch geschaffen werden! Checklisten und Anleitungen zu architektonischem Vorgehen, zur Erstellung architektonisch gegliederter, gut strukturierter, **übergeordneter Anforderungen** lassen sich durchaus formulieren und anwenden.

In Systementwicklungsprojekten architektonisch vorzugehen, bedeutet daher, die übergeordneten Requirements zu erarbeiten, zu gliedern und gut zu dokumentieren – also auch zu **visualisieren** (Abb. 4). Das heißt auch, Anforderunsgmanagement bereits in der Analysephase voll zu integrieren.

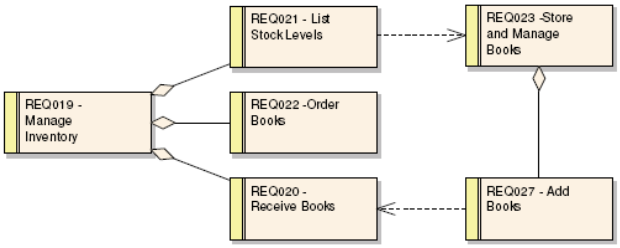
Gleichzeitig ist es erforderlich, allfällig angeführte Lösungsansätze in zu erreichende Ergebnisse, Wirkungen und Ziele *umzugestalten* und diese als qualitative Requirements anzuschreiben, zu sortieren und in saubere Form zu bringen – und alles Lösungsspezifische, das eine unnötige Eingrenzung darstellt, sofort herauszustreichen. Auch dies geschieht vorzugsweise in grafischer Form.

Diese Art der Annäherung an eine Anforderungsaufstellung hat neben den Nachteilen

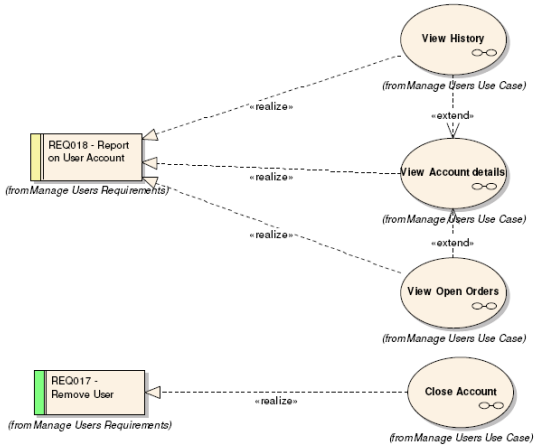
- *Lösungsdenkern* eine Herausforderung bieten zu müssen und
- manchen Teilnehmer des Arbeitskreises überzeugen zu müssen, dass ein Formulieren von Zielen ohne den Weg dorthin zu kennen, durchaus sinnvoll ist

wesentliche Vorteile:

- Motive treten an die Oberfläche – sowohl die Notwendigkeit, zu erreichende Ziele auflisten zu müssen, als auch das implizite Erfordernis, sie messbar zu machen, zeigt automatisch die (vorab zumeist verdeckten) wahren Motive auf
- Voreilig eingebrachte Einschränkungen werden erkannt und zurückgestellt
- Die Kriterien, die das Projekt rechtfertigen, werden umgehend transparent
- Die Teamarbeit ist konfliktfrei – noch weiteifern keine Lösungsvorschläge, sondern es wird ausschließlich abgestimmt, was als funktionales Ergebnis erreicht werden soll/muss



- Diese Sammlung der übergeordneter Anforderungen wird auf Vollständigkeit, Konfliktfreiheit, hinreichende Unterteilung, usw. geprüft und im (Kern-)Team abgesegnet
- Ab diesem Zeitpunkt ist das Einbringen von Lösungsansätzen freigegeben. Jetzt ist es ein Leichtes, vollkommen unkonventionelle Lösungen zu erkennen, die (auch) die Anforderungen erfüllen
- Alle Lösungsvorschläge werden an den übergeordneten, architektonischen Anforderungen auf Übereinstimmung geprüft. Es entstehen keine Konflikte, weil die Beitragenden selbst die Verträglichkeit mit den Anforderungen beurteilen (Abb. 5). Es gilt die einfache Regel: Alles, was aus Lösungsvorschlägen einzelnen Anforderungspunkten nicht zugeordnet werden kann, ist entweder (unnötiger) Luxus oder es sind Ergänzungen, z.B. klassifiziert als „nice to have" nachzutragen. Bleiben Punkte unerfüllt, muss der



Lösungsvorschlag ergänzt oder verworfen werden.
- Möglicherweise werden noch Ergänzungspunkte oder anderer Änderungsbedarf an den übergeordneten Requirements erkannt, diese müssen abgestimmt und sauber eingearbeitet werden. Bereits geprüfte Lösungsansätze müssen in diesem Fall nochmals gegengeprüft werden.
- Es ergibt sich eine strukturierte Gliederung der übergeordneten Anforderungen, der alles Weitere zugeordnet wird. Diese Dokumentation kann von Entscheidungsträgern, vom Management gelesen, verstanden und getragen werden kann.
- Eine vollständige und nachvollziehbare Einbettung in die Systemmodellierung bleibt möglich

Sowohl für die **Visualisierung** dieser übergeordneten Requirements als auch der später nachfolgenden, aus der Realiserungsschicht oder der Implementierungsschicht stammenden, verfeinernden Requirements bietet sich die **Grammatik der UML für Klassen** zur Übernahme in stereotyper Form an. Zwischen einzelnen Requirements können die Beziehungen *Aggregation, Komposition bzw. Verschachtelung, <<deriveRqt>>, <<refine>>, Realisierung oder <<satisfy>> und Abhängigkeit (Dependency)* auftreten. Zu Testfällen bietet sich die Beziehung *<<verify>>* an. Die UML-Grammatik mit ihren Symbolen reicht hier völlig aus, um alle denkbaren Beziehungen und Abhängigkeiten zwischen Requirements in Diagrammen grafisch und im Modell logisch darstellen zu können.

Dies ist eine deutlich Absage an abgesetzte, lineare Anforderungslisten in Textform, eine Wechsel hin zu Anforderungsstrukturen, die logisch in das Systemmodell eingebettet werden, die grafisch top-down in Diagrammen mit Beziehungssymbolen abgebildet werden und die eine vollständige Vernetzung mit Lösungen, Testfällen und Risikobewertungen aufweisen. Je nach Rolle und Aufgaben des Betrachters können sie gemeinsam genutzt werden – und in unterschiedlicher Tiefe nachvollzogen werden.

Weitere Anforderungsdetails: Requirements müssen in mehreren Dimensionen klassifizierbar sein: Status, Phasenzuordnung, Requirement-Kategorie, Ownership, Dringlichkeit, Komplexität, Prüfstatus, usw.. Hier ist einerseits wichtig, dass die Klassifikationslisten gestaltbar, d.h. erweiter- und änderbar sind und andererseits weitere Klassifikationsdimensionen durch geeignete Mechanismen wie z.B. UML-TaggedValues durch den Anwender hinzufügbar sind.

Verlinkungsmöglichkeiten mit einem Risikomanagement-Funktionsblock, einem Metrics- und einem Ressourcen-Zuordnungsblock werden sinnvoll sein. Versionierung und Außerkraftsetzung einzelner Requirements, ohne sie aus der Dokumentation entfernen zu müssen, sind zusätzlich zu fordern. Eine Nachvollziehbarkeit von Änderungen, Such- und Filterfunktionen werden benötigt, dies betrifft aber ohnehin das Modellierungswerkzeug in seiner Gesamtheit – Anforderungsmanagement ist ja nur ein Funktionsteil des Modellierungswerkzeuges.

Die Realisierungs- and Abhängigkeitsbeziehungen müssen zu weiteren Elementen des Modells (Use Cases, Abläufe, Statemodels, OOP-Klassen, usw.) herstellbar sein, sodass die Verkettungen mit der weiteren Analyse, mit dem Design und der Implementation darstellbar, visualisierbar, und abfragbar werden. Reports hierzu müssen abgerufen werden können, einschließlich Teststand und Implementierungsgrad verketteter Elemente. Eine abrufbare, automatische Darstellung in Form einer Beziehungsmatrix ist von Vorteil.

Was ist die **Schlussfolgerung**? Abgesehen von reinen Umsetzungsprojekten mit einer fixen Lösungsvorgabe ist architektonisches Vorgehen im Anforderungsteil – unabhängig von der projektumsetzenden Vorgehensweise – immer die beste Wahl. Die Visualisierung von Requirements in Diagrammform stellt die meist komplexen Zusammenhänge vereinfachend, leicht lesbar, aber dennoch richtig dar. Erst dadurch wird ein strukturierter, nachvollziehbarer und im Projekt lebbarer Top-Down-Ansatz möglich, der kontinuierlich den Blick auf die Zusammenhänge mit den zu erzielenden Wertschöpfungen sichert. Zusätzlich wird das Projektrisiko verkleinert, Auftraggeber und Auftragnehmer haben sich auf die unterstellten Zusammenhänge verständigt und entnehmen den Projektstand aus einer umfassenden, vernetzten und aktuellen Projektdarstellung.

Enterprise Architect von Sparx Systems setzt die skizzierten Anforderungen um, ohne Sie in der Wahl ihres Vorgehensmodells im Projekt festzulegen oder einzuschränken. Sie können diesen Funktionsumfang nutzen, egal, ob Sie iterativ vorgehen, Rapid Prototyping betreiben oder andern Prozessmodellen folgen.

Sollten Sie dennoch - trotz der Vorteile der internen Anforderungsverwaltung des Enterprise Architects - externe Requirements Management Systeme einsetzen wollen, stehen Ihnen mehrere Mechanismen zur Verfügung, diese anzubinden oder sie auch voll zu integrieren.

Noch Fragen? - Wenden Sie sich bitte an sales@sparxsystems.at oder besuchen Sie www.sparxsystems.at .

Über den Autor

Ing. Dietmar Steinpichler hat langjährige, praktische Methodenerfahrung in „architektonischem" Projektvorgehen als selbständiger Softwaredienstleister und Berater. Als Mitarbeiter von mobilkom austria ag hat er als „Solution Architect" mehrere Großprojekte mit diesem Ansatz analysiert, spezifiziert, projektleitend betreut und am Lösungsdesign mitgearbeitet. Als Callcenterspezialist war er an der Entwicklung einer Programmiersprache für CRM-optimiertes Inbound-Call-Routing spezifizierend beteiligt. Derzeit ist Dietmar Steinpichler als Senior Consultant und Trainer für SparxSystems Software GmbH in Europa tätig.
