Vortragsreihe „Software Engineering for Everyday Business“

Software Engineering & Software Prozesse

Dietmar Winkler, Michael Pernkopf

Technische Universität Wien
Institut für Softwaretechnik und Interaktive Systeme

dietmar.winkler@qse.ifs.tuwien.ac.at
http://qse.ifs.tuwien.ac.at

Inhalt

- **Teil 1a: Einführung in die Softwareentwicklung (ca. 60‘)**
  - Typisierung von Projekten
  - Erfolgs/Misserfolgsfaktoren in der Softwareentwicklung
  - Rollen in der Softwareentwicklung
- **Teil 1b: Life-Cycle Phasen (ca. 60‘)**
  - Softwarelebenszyklus
  - Ausgewählte Aspekte in den jeweiligen Phasen
  - Grundlegende Softwareprozesse in Anlehnung an den Life-Cycle Prozess.
- **Teil 1c: Ausgewählte Softwareentwicklungsprozesse (ca. 60‘)**
  - V-Modell XT
  - Rational Unified Process
  - Agile Softwareentwicklung

Vortragsreihe „Von Software Engineering bis Qualitätssicherung“

Software Engineering & Software Prozesse

Teil 1a: Einführung in die Softwareentwicklung

Dietmar Winkler, Michael Pernkopf

Technische Universität Wien
Institut für Softwaretechnik und Interaktive Systeme

dietmar.winkler@qse.ifs.tuwien.ac.at
http://qse.ifs.tuwien.ac.at

Inhalt

- **Teil 1a: Einführung in die Softwareentwicklung (ca. 60‘)**
  - Typisierung von Projekten
  - Erfolgs/Misserfolgsfaktoren in der Softwareentwicklung
  - Rollen in der Softwareentwicklung
- **Teil 1b: Life-Cycle Phasen (ca. 60‘)**
  - Softwarelebenszyklus
  - Ausgewählte Aspekte in den jeweiligen Phasen
  - Grundlegende Softwareprozesse in Anlehnung an den Life-Cycle Prozess.
- **Teil 1c: Ausgewählte Softwareentwicklungsprozesse (ca. 60‘)**
  - V-Modell XT
  - Rational Unified Process
  - Agile Softwareentwicklung

Motivation und Zielsetzung

- Software-Herstellung ist nicht-trivial und erfordert professionelles Herangehen.
  → Software Engineering.
- Unterschiedliche Projekte erfordern unterschiedliche Vorgehensweisen.
  → Software Prozesse & Vorgehensmodelle.

- Welchen Einfluss hat die Projektgröße?
- Software muss nicht immer selbst entwickelt werden – wann soll man ein Produkt kaufen, wann selbst entwickeln?
- Welche Kriterien sind relevant, damit ein Projekt erfolgreich wird?
- Warum scheitern viele Projekte?
- Welche Rollen sind im Software Engineering relevant?

Projekttypen

Verschiedene Anforderungen erfordern unterschiedliche Lösungsansätze.

Projekttyp	Anforderungen	Beispiel
Kommerzielle Software	Benutzbarkeit, Verfügbarkeit, Support	Datenbanktransaktionen
Embedded / Real-time Systems	Zeitgesteuert, safety & security, Echtzeitanforderungen	Handy, ABS-System, Liftsteuerung
Wissenschaftliche Software	Rechengenauigkeit, Korrektheit, Zuverlässigkeit	Medizinische oder Luftfahrtprogramme
Computerspiele	Benutzbarkeit, Funktionalität, Effizienz	
Web Anwendungen	Benutzbarkeit, Sicherheit, Verfügbarkeit	Web Shops

Unterscheidungsmerkmale

Merkmal	Messbare Attribute	Beispiele
Projektgröße	Anzahl der beteiligten Personen	Personen: 50
Projektdauer	Anzahl der Tage / Wochen	Dauer: 52 Wochen
Verwendete Technologie	Art, Anzahl und Alter der Technologie	Art: Scripting Alter: 4 Jahre Anzahl: 5
Komplexität	Anzahl der Klassen, Module, Datenbanken; Verwendete Technologie Lines of Code (LOC)	Datenbanken: 2 Klassen: 42 LOC: 30k

Projektgrößen

Projektgröße	Kriterien	Beispiele
Light	Bis zu 6 Personen Personen-Monate (PM): 0-8 Anzahl Technologien: <5	Rechenprobleme Algorithmen
Medium	10-30 Personen 9-24 PM Technologien: 5-12	Buchhaltung Lagerverwaltung
Heavy	50-100 Personen 25-45 PM 12-20 Technologien	Compiler Datenbank
Super Heavy	Ab 100 Personen >45 PM >20 Technologien	Raumfahrt Atomkraftwerk elektronische Börse

Was ändert sich bei Zunahme der Größe?

- Steigende Komplexität der Anwendung (z.B. erweiterte Funktionalität)
- Zunehmender Bedarf an Flexibilität der Lösungen.
- Erhöhter Bedarf für organisatorische und planerische Aktivitäten.
- Steigender Bedarf an Kompetenzen innerhalb des Teams.
- Klare Definition der Kommunikationsstrukturen.
- Erhöhter Testaufwand.
- Erhöhter Dokumentationsaufwand zur Nachvollziehbarkeit der Projektprozesse (speziell im Hinblick auf System-Schnittstellen).
- Regelungen für Versionskontrolle, Konfigurations- und Änderungsmanagement.

Software Beschaffung – Kauf oder Entwicklung?

	Kauf	Abänderung	Neuerstellung
Entspricht den Anforderungen	Ungefähr (oft viele ungenützte Funktionen oder einige fehlende Funktionen)	Oft keine exakte Anpassung möglich → Beschränkung durch technische Grenzen	Genau
Änderbarkeit	Schwierig, da technische Details oft nicht transparent sind	Abhängig vom Ursprung des Ausgangsprodukts: - Eigenentwicklung: leicht änderbar - Fremdentwicklung: schwer änderbar	Gut möglich (Dokumentation der Entwicklung und technische Transparenz vorhanden)
Preis	Nach Anforderung & Vereinbarung	Eigene IT: kalkulierbar Fremde IT: kostenintensiver	Teuer

Software Beschaffung – Kauf oder Entwicklung?

Kauf von Software, wenn

- die gewünschten Anforderungen weitgehend erfüllt werden.
- Die individuelle Anpassung (leicht) möglich ist.
- Die Kosten niedriger sind als bei Eigenentwicklungen.
- Ausführliche Dokumentation vorhanden ist.
- Guter Support bei Problemen und Fragestellungen geboten wird.

Andernfalls

- Suche nach alternativen Lösungen.
- Eigenentwicklung.

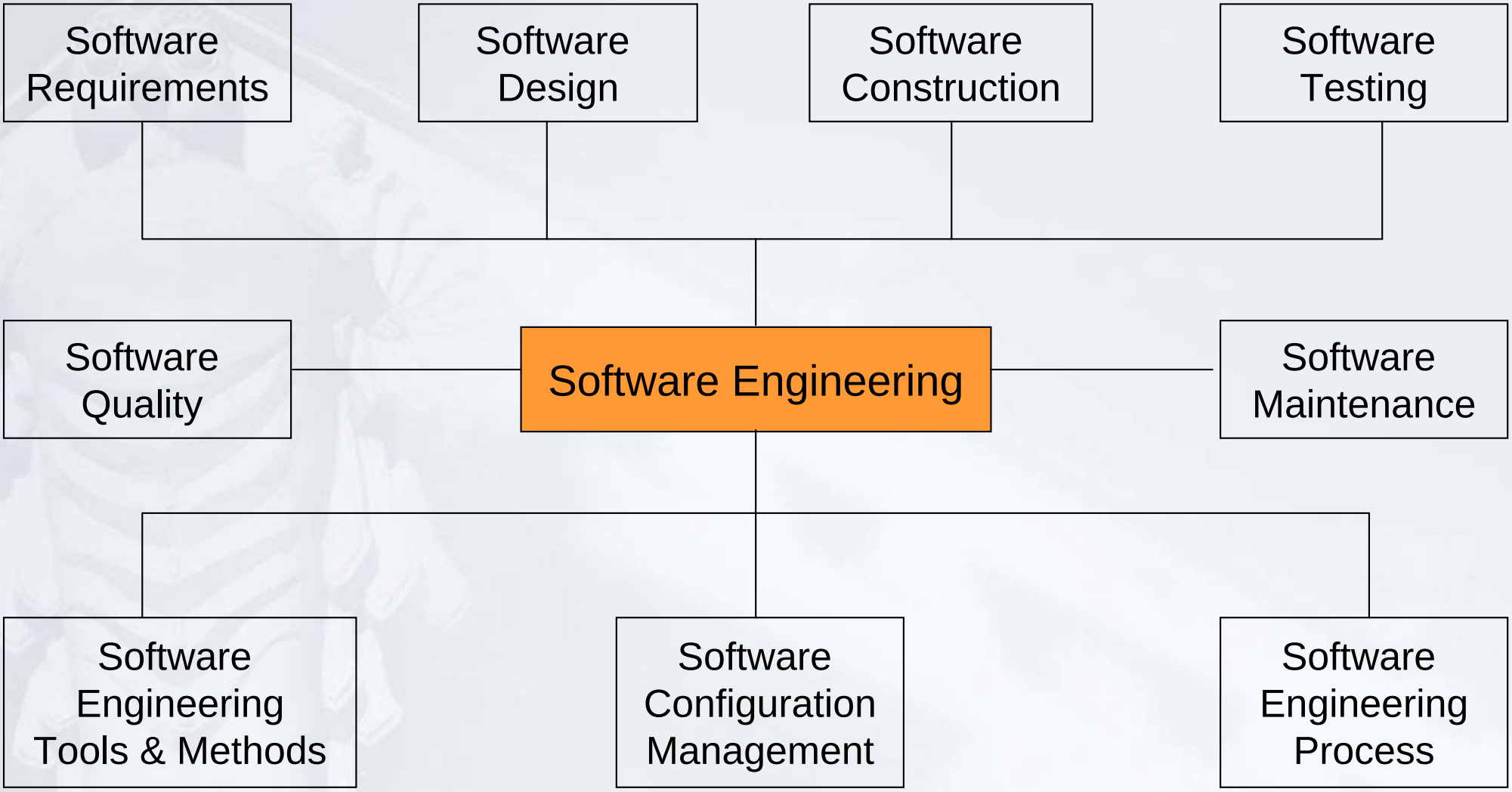
Faktoren für ein erfolgreiches Projekt

- Einbringung und Berücksichtigung der Anwender.
- Adäquates Projektmanagement: nicht zu viel aber auch nicht zu wenig.
- Anforderungen müssen eindeutig beschrieben, realisierbar und auch überprüfbar sein.
- Flexibler, realistischer Projektplan mit Spielräumen (Berücksichtigung möglicher Verzögerungen).
- Realistische Kostenschätzung und Budget.
- Angemessene Ziele.
- Schlüsselteammitglieder verfügen über ausreichend Projekterfahrung.
- Gute Teamarbeit, funktionierende Kommunikation im Team.

Top Ten der Gründe für Projektmisserfolg

1. Unvollständige Anforderungen.
2. Anwender nicht involviert.
3. Zu wenig Ressourcen.
4. Unrealistische Zeit- und Kostenpläne.
5. Keine Managementunterstützung.
6. Häufige Änderung der Anforderungen
7. Qualitätsmängel bei extern vergebenen Komponenten.
8. Qualitätsmängel bei extern vergebenen Aufgaben.
9. Fehlende Planung.
10. Projekt wird nicht mehr benötigt.

Software Engineering Disziplinen



Rollen in der Softwareentwicklung

- Software Engineering is a “Contact Sport” (Boehm, 2002).
- Die Anzahl der Teammitglieder (stakeholder) hängt vom Projekttyp und der Projektgröße ab.
- Unterschiedliche Rollenzuteilung in Abhängigkeit von der Unternehmensgröße (KMUs vs. “große” Unternehmen)
- Konflikte resultieren aus unterschiedlichen Sichtweisen der verschiedenen Stakeholder.

Beispiele für typische Rollen in der Softwareentwicklung:

- Kunde und Anwender: Projektinitiierung und Anwendung des Systems; Zusammenarbeit in grundlegenden Entscheidungen (Anforderungen).
- Projektleiter / Produktmanager: Planung, Koordination, Controlling und organisatorische Aufgaben in unterschiedlichen Projekten.
- Ausführende Rollen: Analyst, technischer Architekt, Qualitätssicherer, Tester, Technical Writer.

Rollen in Abhängigkeit der Projektgröße

Projektgröße	Rollen	Warum?
Light	Kunde, Management, Programmierer, Tester	Besteht meistens aus nur einer Person die alle Rollen in sich vereint.
Medium	Kunde, Projektleiter, Analytiker, Programmierer, Tester, Controller.	Projekt umfangreicher Zusammenarbeit mehrerer Personen notwendig
(Super) Heavy	Kunde, Management, wirtschaftlicher und technischer Projektleiter, Gruppenleiter, Analytiker, Controller, Programmierer, Tester, Qualitätssicherer.	Projekt für Einzelne nicht mehr überschaubar, Aufteilung in Gruppen mit Struktur des Ganzen

Rollen in “kleinen” Projekten

- Unterschiedliche Rollen konzentrieren sich auf einzelne Personen
  z.B.:
  - Der Programmierer testet sein Programm selbst.
  - Der Projektmanager ist gleichzeitig auch der Projektleiter.

- Erforderliche Rollen:
  - Kunde: Projektinitiierung (als Vertragspartner);
    Definition grundlegender Produktanforderungen.
  - Management: Planung der Vorgehensweise und Überwachung des Fortschritts.
  - Programmierer: Implementierung der Softwarelösung.
  - Tester: Bewertung und Prüfung des Produktes auf Erfüllung der Anforderungen.

Rollen in “großen” Projekten

- Ein sehr großes Team entwickelt ein Produkt für eine große Anzahl von Anwendern.
- Beispiel: Web-Shop

- Erforderliche Rollen
  - Kunde: Projektinitiierung (als Vertragspartner); Definition grundlegender Produktanforderungen.
  - Projektleiter: Projektkoordination zwischen unterschiedlichen Stakeholdern.
  - Teamleiter: Verantwortlich für ein Teil-Projekt inkl. Reporting.
  - Analyst: Analyse der Systemanforderungen und Modellierung dieser Anforderungen.
  - Designer: User Interface Design.
  - Programmierer: Implementierung der Requirements, Code integration.
  - Qualitätssicherer: Überprüfung der Produkte und Definition der Qualitätsattribute.
  - Tester: Überprüfung und Test des Produktes auf Erfüllung der Anforderungen.
  - Dokumentierer: Projekt / Produktdokumentation.

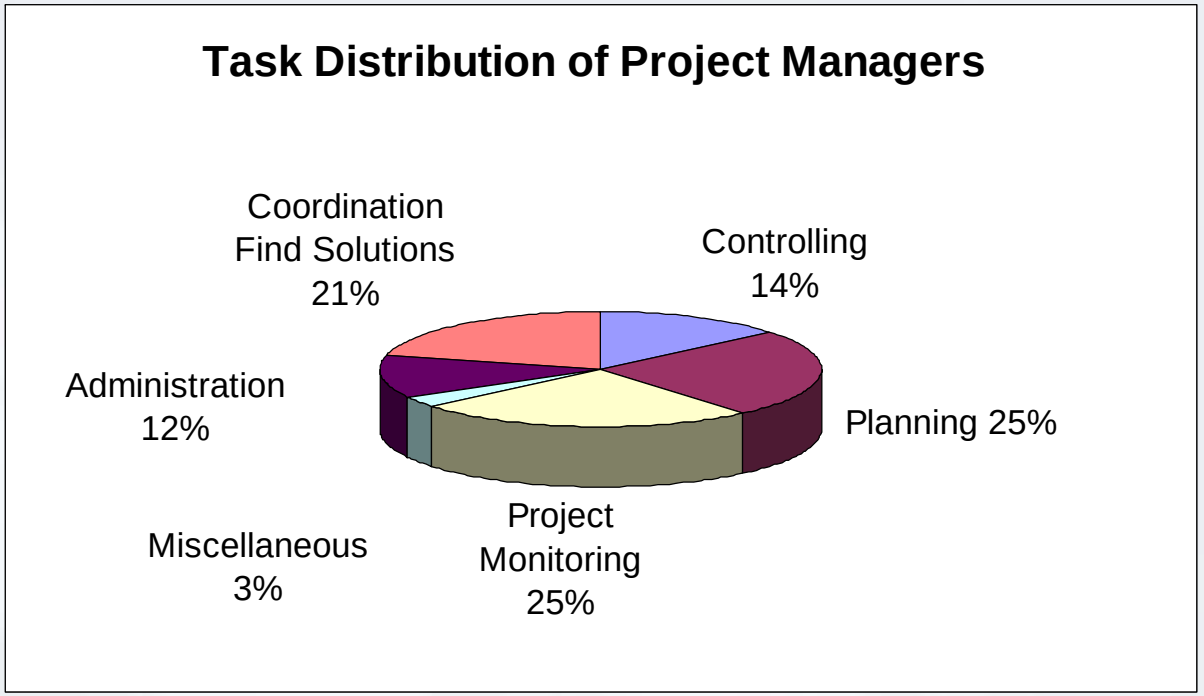
Funktionale Rollen - Anwender

- **Aufgaben**
  - Beteiligung bei der Erstellung der fachlichen Anforderungen.
  - Identifikation der DV-technisch zu realisierenden Funktionen.
  - Festlegung der organisatorischen Regelungen für die Nutzung des Systems.
  - Mitarbeit bei Prüfungen und Abnahmen.

- **Geforderte Kenntnisse und Fähigkeiten**
  - Kenntnisse über die Anwendung und Einsatzgebiete des Systems.
  - Kommunikationsfähigkeit mit dem Systemanalytiker, Systemdesigner und Systembetreuer.

Funktionale Rollen - Projektmanager

- Der Projektmanager unternimmt alles, um das Projekt in Gang zu halten.
- Er übernimmt keine Durchführungsaufgabe selbst, er arbeitet zwar an der Lösungsfindung mit, die Umsetzung selbst erfolgt jedoch durch die Projektmitarbeiter.
- Der Projektmanager ist für das gesamte Projekt der Hauptverantwortliche.



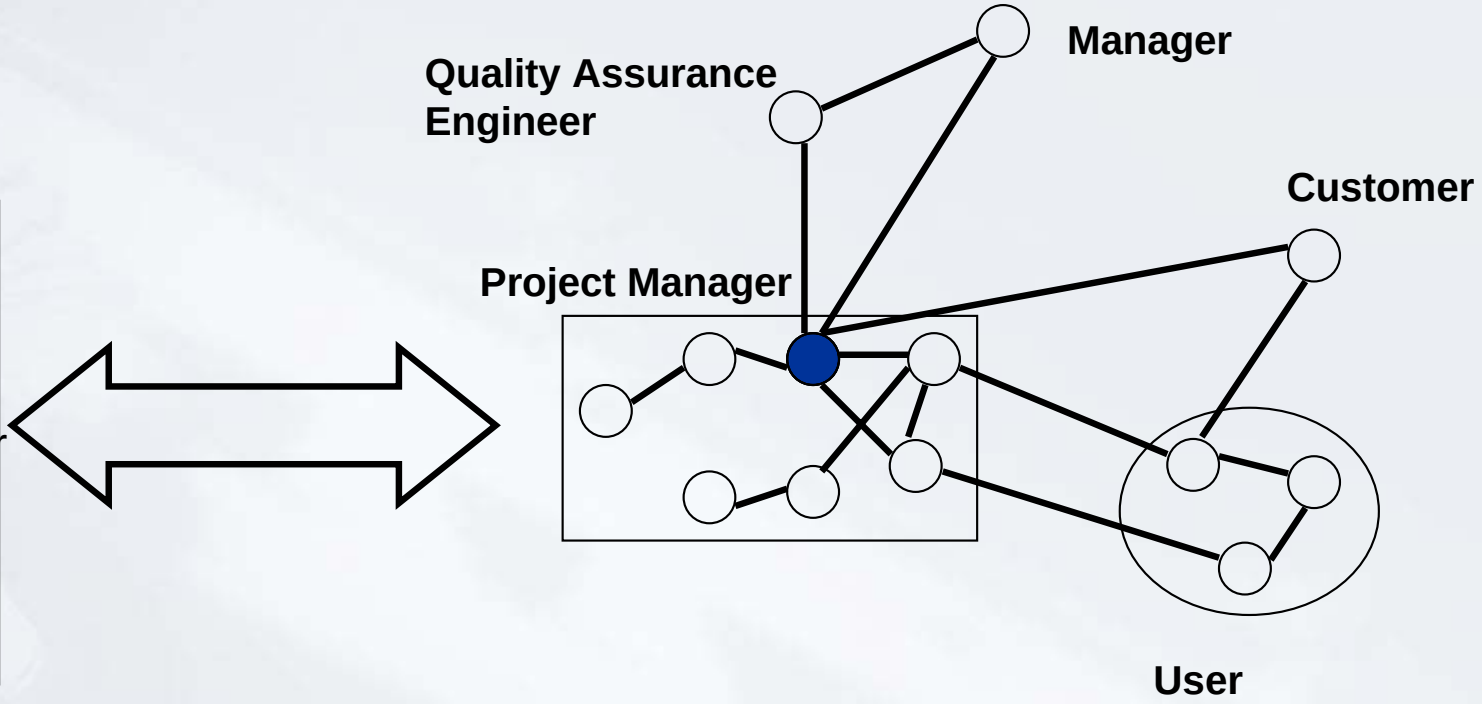
Funktionale Rollen – Ausführende Einheiten

- **Aufgaben**
  - Modellierung des Systems.
  - Definition der Qualitätsanforderungen.
  - Realisierung der Module/Datenbanken + Mitarbeit bei Prüfungen und Abnahmen.
  - Erstellung von QS-Plan, Prüfplan und Prüfspezifikation.
  - Bewertung eines Prüfgegenstands anhand der vorgegebenen Prüfspezifikation.
  - Projektdokumentation.

- **Geforderte Kenntnisse und Fähigkeiten**
  - Kenntnis über Anwendung und Einsatzgebiete des Systems.
  - Technisches Verständnis.
  - Kenntnis über Methoden und Werkzeuge.

Kommunikation in SE Projekten

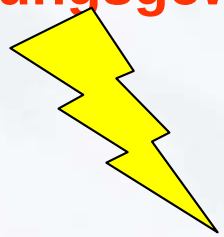
Working Group:
Analysts and domain experts,
Systems architects, developer,
User Interface Designer, Tester
Technical Writer, Tool Expert,
Database expert



Konflikte zwischen Rollen

- Oft ist es schwierig, die Erwartungen der Stakeholder unter einen Hut zu bekommen, da sie unterschiedliche Ziele verfolgen und somit in Konflikt geraten [Duncan, 1996].

- Beispiel Matrixorganisation:
  - Der Projektleiter benötigt mehr Personen zur Fertigstellung des Projekts.
  - Der HR-Manager hat Interesse daran, dass möglichst wenig Personen im Einsatz sind und teilt dem Projekt keine zusätzlichen Kräfte zu.
  - **Aufgrund der gleichberechtigten Entscheidungsgewalt in einer Matrixorganisation kommt es zum Konflikt.**



Stärken/Schwächen – KMUs vs. Großbetriebe

	KMUs	Großbetriebe
Stärken	▪ Flexibilität ▪ Persönliche Kenntnis von Stärken/Schwächen im Projektteam ▪ Hoher Teamgeist	▪ Finanzielle und personelle Reserven für langfristige Investitionen in Personal ▪ Karrierewege im Haus ▪ Ausgleich für Fluktuation
Schwächen	▪ Geringere Investitionsmöglichkeiten ▪ Fokus auf kurzfristige operative Ziele ▪ Wenig robust gegen Durststrecken	▪ Oft starre Prozesse

Skill-Management – KMUs vs. Großbetriebe

	KMUs	Großbetriebe
Personalentwicklung	▪ Systematische Personalentwicklung fehlt häufig	▪ Training, Organisation ▪ Personalentwicklungsplan ▪ Schulungspläne
Strategische Reserven		▪ Vernetzungsplattform ▪ Wissensprofile
Möglichkeiten	▪ IT-Infrastruktur bestehend aus Insellösungen	▪ Kommunikation und Informationsplattformen ▪ Wissensdatenbanken

[Bellman et al, 2002, Barske et al, 1999, Antoni et al, 2001]

Zusammenfassung

Projekttypen

- Verschiedene Anforderungen erfordern verschiedene Lösungsansätze.
- Mit der Projektgröße ändern sich viele Faktoren des Projektes
  z.B. Komplexität, Bedarf and Planung, Organisation, etc.
- Die Eigenentwicklung ist nicht immer die beste Lösung

Rollen

- Unterschiedliche Rollen je nach Projektgröße
- Konflikte komplizieren Entscheidungen.
- Unterschiedliche Handhabung in KMUs und Großbetrieben.

Referenzen

- Antoni C. H. , Sommerlatte T. , Wissensmanagement 2001, Symposion Publishing GmbH.
- Barske, Gerybadze, Hünninghausen, Sommerlatte Das innovative Unternehmen 1999, Symposion Publishing GmbH.
- Bellmann M., Krcmar H., Sommerlatte T. , Wissensmanagement 2002, Symposion Publishing GmbH.
- Boehm B.: „Software Engineering is a Value-Based Contact Sport“, IEEE 2002.
- Boehm B.: Software Risk Management: Principles and Practices, IEEE Software 8(1), pp. 32-41, 1991.
- Duncan William R.: A Guide to the Project Management Body of Knowledge, 1996.
- IESE http://www.iese.fhg.de/VModell/SE/SERollen.html.
- Sommerville, Ian: „Software Engineering“, 6th Edition, Addison Wesley, 2001, ISBN: 020139815x.
- SWEBOK: Guide to the Software Engineering Body of Knowledge, http://www.swebok.org, 2005.
- Zopf S.: Lehrveranstaltung: Fortgeschrittene Aspekte des Qualitätsmanagements, Software-Projektmanagement, 2003.

Vielen Dank für die Aufmerksamkeit !

Kontakt: Dipl.-Ing. Dietmar Winkler, Michael Pernkopf

Technische Universität Wien
Institut für Softwaretechnik und Interaktive Systeme
Favoritenstr. 11/188, A-1040 Vienna, Austria

dietmar.winkler@qse.ifs.tuwien.ac.at
http://qse.ifs.tuwien.ac.at