Entwicklungsmethoden

Prof. Dr. Josef M. Joller
jjoller@hsr.ch

TEAMS

Übersicht

Team Organisation

Demokratische Team Organisation

klassische Chief Programmer Team Organisation

Mischformen

Synchronize-and-Stabilize Teams (Microsoft)

Extreme Programming Teams

Programmier Team Organisationen

Ein Produkt muss in drei Monaten realisiert werden.
Der geplante Aufwand beträgt aber 1 Personenjahr.

Lösung

- Falls ein Programmierer ein Jahr benötigt, benötigen vier Programmierer drei Monate

Nonsense

- mit vier Programmierern benötigen Sie vermutlich nur wenig weniger lang als ein Jahr
- die Qualität des Produktes wird tiefer liegen als im Falle einer Einzelarbeit

Task Sharing

Beim Erdbeerenpflücken kann man die Produktivität des Teams durch zusätzliche Arbeiter linear steigern, da jeder Arbeiter für sich abeitet.

Falls eine Frau 9 Monate für die Austragung eines Babies benötigt, svhaffen es drei Frauen in drei Monaten nicht.

Lehre

es gibt viele unteilbare Tätigkeiten,
Tätigkeiten, welche nicht einfach linear aufgeteilt werden können

Team Organisationsformen

Beispiel:

- zwei Programmierer arbeiten je an einem Modul mA and mB.

Was könnte alles schiefgehen?

- Im schlimmsten Fall arbeiten beide ohne es zu wissen je am gleichen Modul
- ursprünglich wurde festgelegt, dass Modul A fünf Parameter besitzt;
  bei der Realisierung vergass der eine Programmierer einen Parameter oder wechselte den Datentyp

Alle diese Probleme haben nicht mit den Skills der Programmierer zu tun

- Teamorganisation ist eine Managementaufgabe

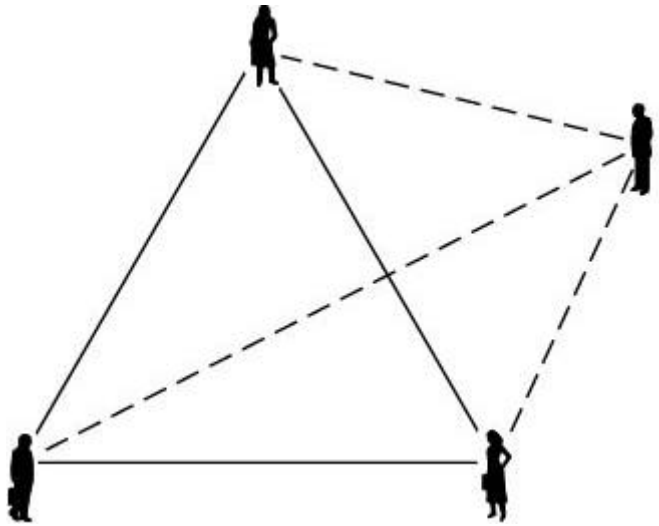
Kommunikations Probleme

Beispiel

- bei drei Programmierern sind nur drei Kommunikationswege möglich.
- aus Zeitgründen wird ein weiterer Programmierer ins Team eingeschleust

Konsequenz

- komplexere Kommunikation



Kommunikations Probleme

Die drei im Team aktiven Mitarbeiter müssen dem neuen

- erklären, wo man steht, welche Fehler vorhanden sein könnten
- helfen das System zu verstehen

Brooks's Gesetz

- Wenn immer man in einem Projekt eine Verspätung durch zusätzliches Personal aufholen möchte, führt ein zusätzlicher Mitarbeiter zu weiteren Verzögerungen.

Team Organisation

In der Software Entwicklung benötigt man immer Teams

- um die SW effizient implemenieren zu können

zwei grundsätzliche, extreme Team Organisationen

- Demokratisches Team (Weinberg, 1971)
- Chief Programmer Team (Brooks, 1971; Baker, 1972)

Demokratische Teams

Grundprinzip—”egoless programming”

Der Programmierer identifiziert sich 100% mit seinem Code

- er betrachtet seine Module als “sein Baby” / “ein Teil von mir”

Nebeneffekt

- ein 🕷 (Bug) wird eher geleugnet als akzeptiert
- “meine Module sind perfekt”

Demokratische Teams

Mögliche Lösung des Konflikts

Egoloses Programmieren

- Wertesystem der Programmierer verändern
- Team motivieren, die Fehler (auch der andern) zu finden
- das Team als Ganzes muss ein gemeinsames Wertesystem und Ziel festlegen und darauf hin arbeiten

Teams aus egolosen Programmieren bezeichnet man als Demokratische Projektteams

Stärken des Demokratischen Teams

Extrem hohe Produktivität (gemeinsame Ziele,”wir” Gefühl)

Funktioniert nur selten

Funktioniert am Besten bei Forschungsprojekten

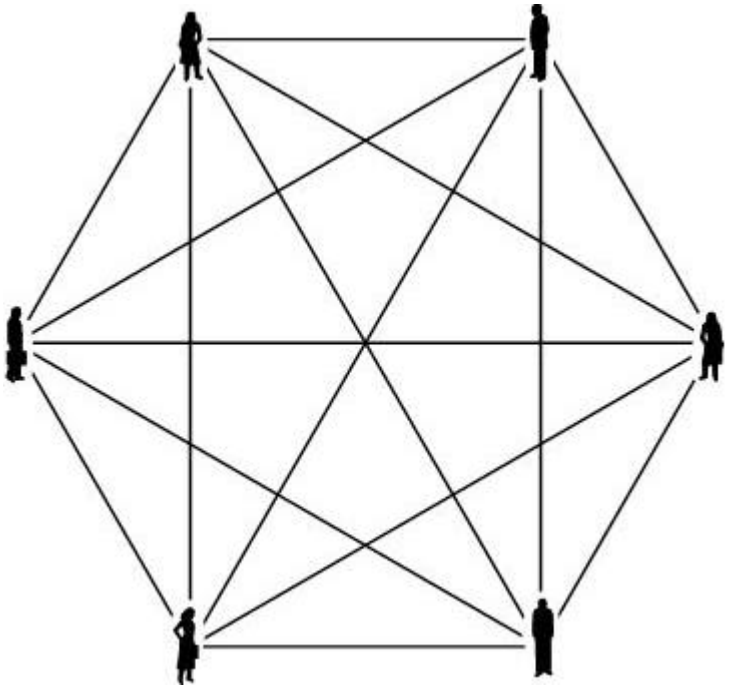
Problem:

- Demokratische Teams funktionieren lediglich, wenn sie spontan entstehen.

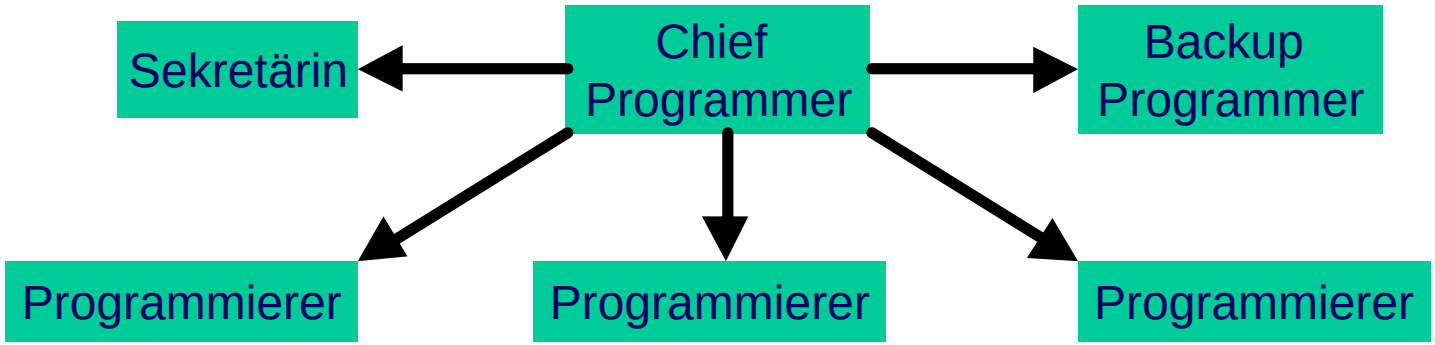
Chief Programmer Teams

Betrachten wir ein 6-Personen Team

- 15 x 2-Person Kommunikation
- total 57 mögliche 2-, 3-, 4-, 5- und 6-Personen Gruppen
- das Team kann unmöglich 6 Personen-Monate in 1 Monat abwickeln



Chief Programmer Team



6 Programmierer, aber nur 5 Kommunikationskanaäle

Klassisches Chief Programmer Team

Grundidee

- Analogie: der Chefarzt / Chirurge muss alles im Griff haben
  - andere (Hilfs-) Chirurgen
  - Anästhesisten
  - Schwestern
  - andere Spezialisten

Schlüsselaspekte

- Spezialisierung
- Hierarchie

Klassisches Chief Programmer Team

Chief Programmer

- ist ein erfolgreicher Manager und hochbegabter Programmierer
- ist verantwortlich für die Architektur
- verteilt die Programmieraufgaben
- schreibt die kritischen Programmteile selber
- überprüft die Ergebnisse der anderen Teammitglieder
- ist für jede einzelne Programmzeile verantwortlich

Klassisches Chief Programmer Team

Back-up Programmer

- ersetzt den Chief Programmierer falls dieser krank wird, in die Ferien möchte,...
- Er kennt jedes Detail des Projekts und ist genau so kompetent wie der Chief
- führt die Planung der “Black Box” Tests durch

Klassisches Chief Programmer Team

Sekretärin

- hoch qualifiziertes Mitglied des Teams
- verantwortlich für die produktiven Programmteile
  - Quellcode
  - Shell Skripts / JCL
  - Testdaten
- Programmierer liefern ihre Ergebnisse der Sekretärin, die
  - die Programme übersetzt
  - Tests durchführt
  - die Programme linked

Klassisches Chief Programmer Team

Programmierer

- ist für das Programmieren und nur dafür zuständig
- alle anderen Aufgaben werden durch die Sekretärin übernommen.

The New York Times Projekt

Chief Programmer Team Konzept

- erstes Projekt, 1971
- durch IBM
- Erstellen einer umfangreichen DB Anwendung der *The New York Times*

Chief Programmer—F. Terry Baker

The New York Times Projekt

83,000 Programmzeilen (lines of code :LOC)

In 22 Kalendermonaten erstellt: 11 Personenjahre

Nach einem Kalenderjahr war lediglich die Dateiverwaltung vorhanden (12,000 LOC)

Die meisten Programme wurden in den letzen 6 Monaten erstellt

21 Fehler wurden in den ersten 5 Wochen der Akzeptanztests gefunden.

25 weitere Fehler wurden im ersten produktiven Einsatzjahr gefunden

The New York Times Projekt

Die Programmierer erstellten im Schnitt 10’000 pro Jahr und verursachten lediglich einen Fehler!

Die Dateiwartung wurde eine Woche nach Fertigstellung abgeliefert und funktionierte 20 Monate fehlerfrei

Die Hälfte der Unterprogramme umfasste normalerweise lediglich 200 - 400 PL/I Quellcode und wurden in der Regel gleich beim ersten Mal fehlerfrei übersetzt.

Warum war das NYT Projekt so erfolgreich?

Prestige Projekt für IBM

- erster Grosseinsatz von PL/I (IBM Programmiersprache)
- IBM (Headquarter nördlich von NY) setzte die besten Leute ein

Alle Teammitglieder waren sehr hoch qualifiziert

- PL/I Compiler Bauer halfen mit
- JCL Experten halfen bei der Erstellung der Job Control Skripte

Der Erfolg wurde in keinem weiteren Projekt wiederholt

Warum war das NYT Projekt so erfolgreich?

F. Terry Baker

- Superprogrammer
- Spitzenmanager und charismatischer Führer
- er prägte das gesamte Projekt

Stärken der Organisationsform

- sie funktioniert
- mehrere Projekte mit abgewandelter Chief Programmer Organisationsform waren auch efolgreich

Was funktioniert eher nicht?

Chief Programmierer müssen hochbegabt sein, als Linienmanager und als Programmierer

- solche Personen gibt es kaum
- Top-Programmierer UND Linienmanager sind nicht seltener

Back-up Programmierer müssen genau so gut sein, wie der Chief.

- Aber er verdient weniger und steht weniger im Mittelpunkt
- Frustjob für Top-Manager und Top-Programmierer

Sekretärin macht nur Papierjobs, den ganzen Tag

- Software Entwickler hassen Papierjobs

Konsequenz: Chief Programmer Teams sind praxisfern

Varianten der Projektorganisation

Gesucht sind Teamorganisationsformen, welche

- die Stärken der Organisationsformen “Chief Programmer” und “Demokratisches Team” verknüpfen
- Teams mit 20 (oder 120) Programmierern hanhaben kann

Demokratische Teams

- suchen proaktiv Fehler

Chief Programmer Teams

- haben Kommunikationswege und Management im Griff

Teamorganisationsformen

Mögliche Probleme

Der Chief Programmer ist für jede Programmzeile verantwortlich

- er muss bei jedem Review anwesend sein

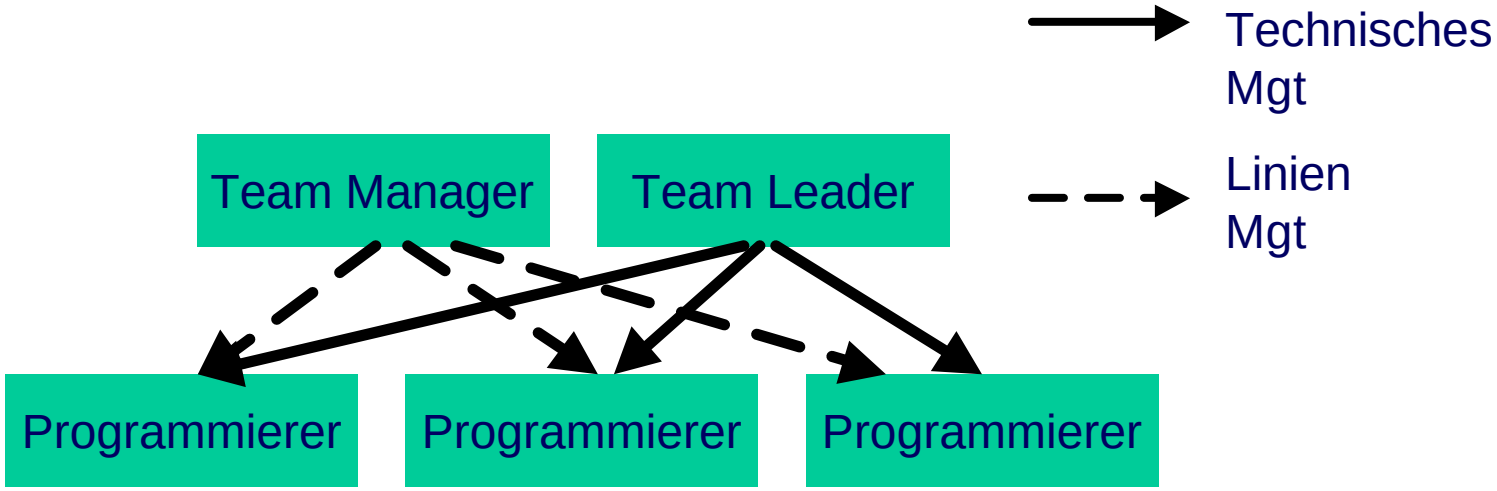
Der Chief Programmierer ist aber auch Team Manager

- er sollte daher die Reviews nicht beeinflussen!

Teamorganisationsformen

Mögliche Lösungen

- Der Chief Programmer muss entlastet werden
- Es ist leichter einen Teamleader als einen Chief Programmer zu finden
- Der Teamleader ist für die technischen Aspekte im Team zuständig



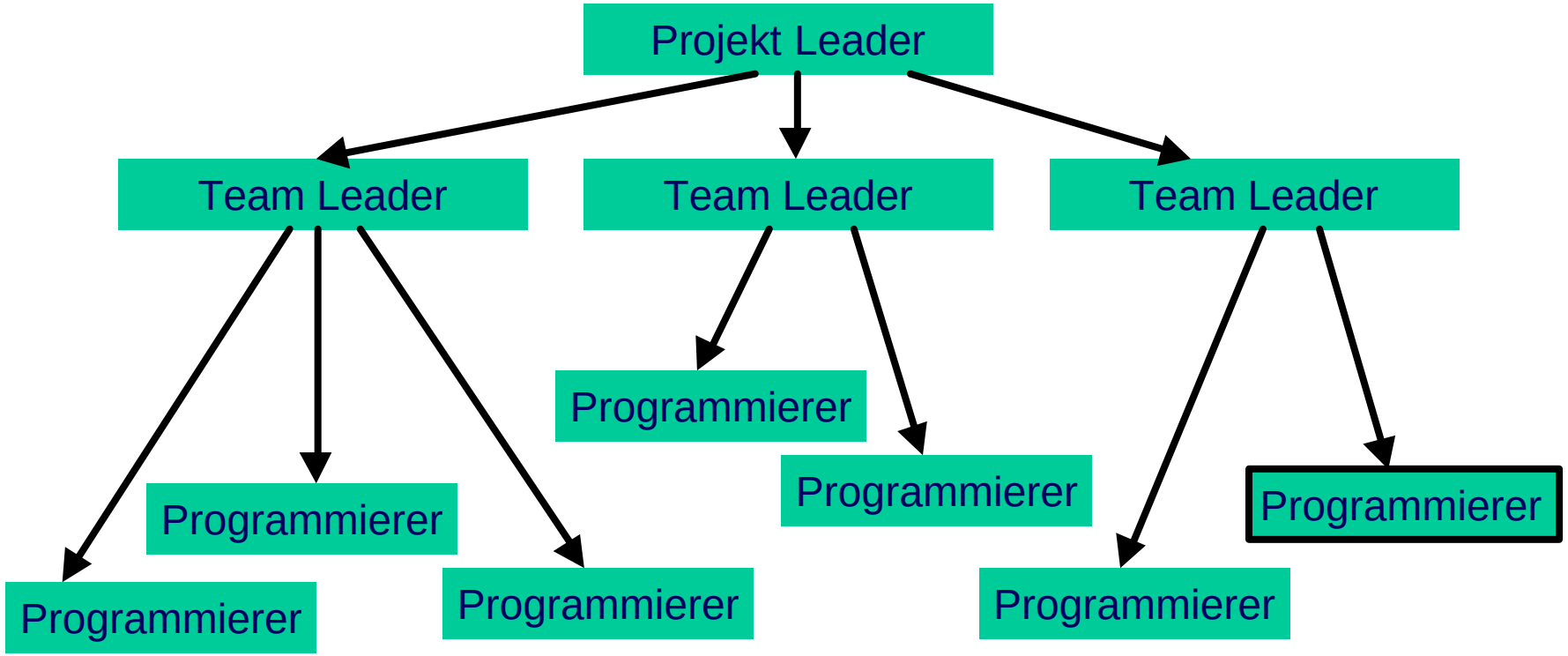
Teamorganisationsformen

Budgets und alle legalen Aspekte und Human Ressource Themen (Ziele, Bonus, ...) werden dem Teamleader abgenommen

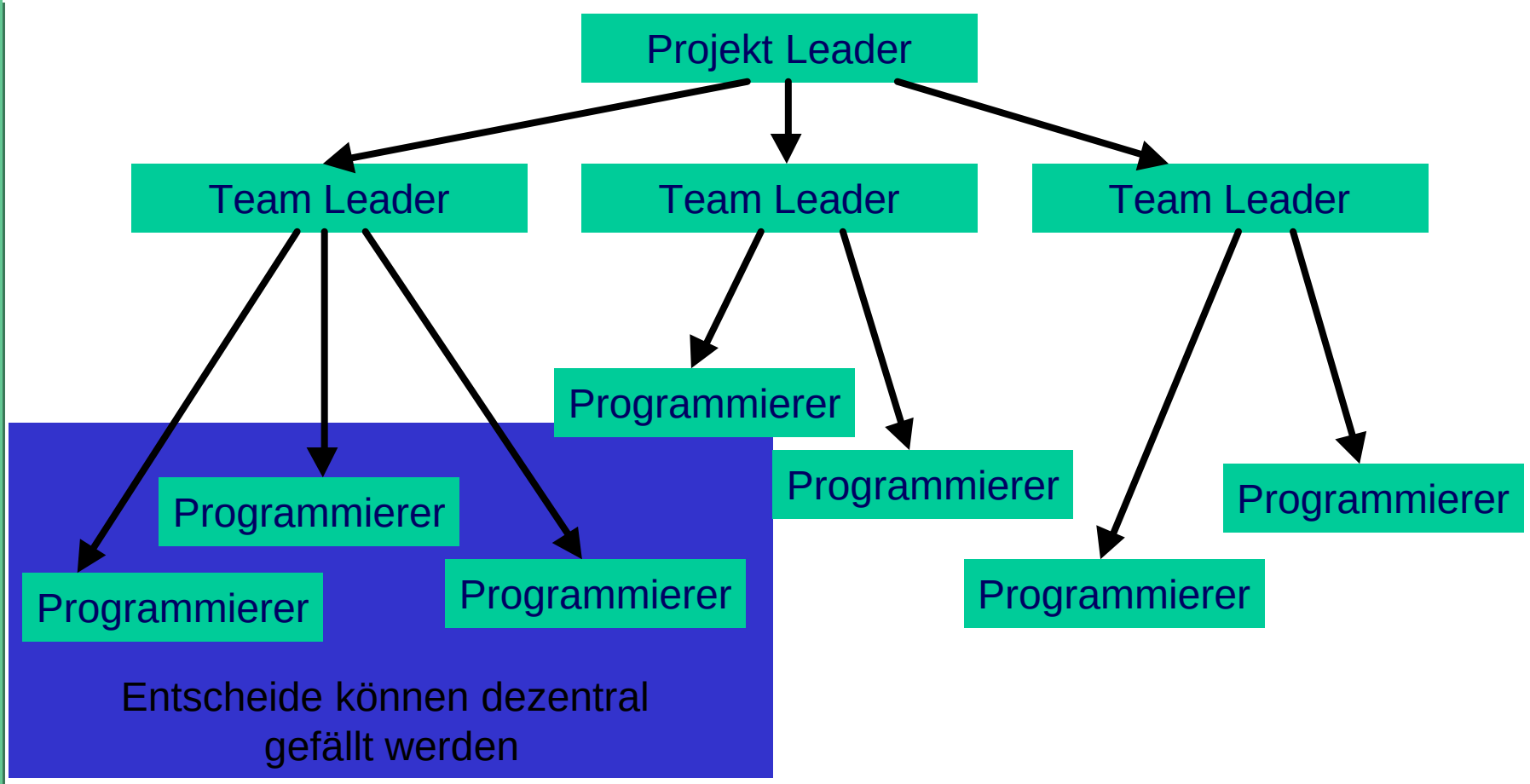
Der Teamleader nimmt an den Reviews teil, der Team-Manager nicht.

Team Managers müssen an den technischen Meetings teilnehmen

Grossprojekte



Grossprojekte



Synchronize-and-Stabilize Teams

Microsoft

Produkte bestehen aus 3 oder 4 sequentiellen Builds / Teilprodukten

Kleine Teams arbeiten parallel

- 3 bis 8 Entwicklern
- 3 bis 8 Testern (arbeiten je mit einem oder mehreren Entwicklern)
- Jedes Team trägt die volle Verantwortung für sein Teilsystem
- Teams können sich ein Teilsystem aussuchen (weitestgehend)

Wege aus dem Hacker Chaos

- eswird täglich synchronisiert
- das Zusammenspiel einzelner Module muss immer funktionieren

Synchronize-and-Stabilize Teams

Regeln

- jedes Teilteam muss sich an die Zeitvorgaben halten, täglich!

Analogie

- die kinder können tun und lassen was sie wollen…
- … aber um 21:00 müssen sie ins Bett!

Ist das Modell allgemeiner einsetzbar?

- Die Organisationsform ist angepasst auf die Microsoft Unternehmensphilosophie
- es liegen zuwenig praktische Beispiele vor (von anderen Unternehmen)

Die Zukunft des XP (Extreme Programming)

- alle Programme werden von je zwei Programmierern erstellt, die an einem gemeinsamen Rechner arbeiten
- “Pair Programming”
  - ein Team Mitglied konzentriert sich auf eher langfristige Aspekte (Architektur)
  - falls der Programmierer ausfällt springt sein Vice ein
  - gute Möglichkeit neue Leute heranzuziehen
  - fast ein egoloses Programmieren (beide helfen sich gegenseitig)

Schlussbemerkungen

Es gibt keine optimale Projektorganisation

Die optimale Projektorganisation ist abhängig von

- dem Produkt
- der Leaderfigur und dessen Visionen
- der Erfahrung der Teammitglieder

Das Thema ist weitestgehend unerforscht

- in der Regel stützt man sich auf die Ergebnisse der Gruppendynamik

Link

Einfacher Jung/Myers-Briggs Typologie Test.

http://www.humanmetrics.com/cgi-win/JTypes2.asp