

Grundlagen des Messens

Tobias Riasanow

Proseminar IT-Kennzahlen und Softwaremetriken

Technische Universität München

Tobias.Riasanow@googlemail.com

Abstract: Von der Frage „Was sind Messungen“, wo begegnet man ihnen im Alltag oder der Wissenschaft bis hin zur Definition formalen Messens gibt dieser Bericht einen Überblick über die Grundlagen des formalen Messens mit besonderem Blickwinkel auf die Softwareentwicklung. Detailliert wird die *Representational Theory of Measurement* betrachtet. Besonderes Augenmerk liegt auf den unterschiedlichen Darstellungsmöglichkeiten der Messskalen. Hierzu zählen die Nominal-, Ordinal-, Intervall- und Verhältnisskala, sowie die absolute Skala. Diese werden genau differenziert sowie charakterisiert und mit Beispielen und Grafiken illustriert.

1 Messen: Was ist das und warum machen wir es?

1.1 Warum misst man?

„**Messen**“, der Mensch tut es täglich. In der Wirtschaft erfasst er z.B. den Preis von Produkten oder Dienstleistungen sowie von Angebot und Nachfrage, in der Medizin dient es zur Diagnose von Krankheiten oder in der Technik zur Positionsbestimmung von U-Booten mithilfe von Sonaren oder zur Ausarbeitung des Wetterberichtes für die nächsten Tage in der Meteorologie. Sogar im Alltag wenden wir schon bei einem Supermarkteinkauf jenes Prinzip des Messens an, wenn wir Preise vergleichen. Doch was genau versteht man unter „Messen“? „*Measurement is a common and necessary practise for understanding, controlling and improving our environment*“¹ [FP97]. Jenes Zitat stammt aus dem Buch „Software Metrics“ von Fenton und Pfleeger, die sich intensiv mit Softwaremetriken beschäftigt haben. Somit kommen einer Messung nicht nur die Funktionen des Verstehens und Kontrollierens zu, sondern auch die Möglichkeit Verbesserung schaffen zu können.

¹ [FP97], Seite 4

Heutzutage versucht der Mensch in allen für ihn relevanten Bereichen nützliche Messungen durchzuführen und sogar Galileo Galilei sagte vor ca. 400 Jahren: „*Miss alles, was sich messen lässt, und mach alles messbar, was sich nicht messen lässt!*“².

So sind im Software-Engineering (SE) potenzielle Messgründe die zeitliche Einordnung der Fertigstellung oder die Beantwortung der Frage, ob das Budget ausreicht. Messcharakteristiken von Software sind z.B. Qualität des Designs oder Vollständigkeit. Durch die Erfassung solcher Charakteristika entsteht vor allem großer Kundennutzen, da kundengerechte Software schließlich nur durch genaue Evaluation möglicher Problematiken entstehen kann.

Somit kann man zusammenfassend sagen, dass ohne Messungen Technologien und vor allem deren Fortschritt nicht ohne Messungen nicht funktionieren sind und das Messen deswegen einen so hohen Stellenwert in der menschlichen Forschung und Arbeit einnimmt!

1.2 Was ist eine Messung?

„*Measurement is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way to describe them according to clearly defined rules*“³ [FP97]. Diese Definition von Fenton und Pflieger versteht den Messvorgang also als den Prozess Objekte der Realität in Zahlen oder Symbole sinngemäß und nach eindeutig festgelegten Regeln zu transferieren. Für uns bedeutet das auch noch, dass nicht Objekte oder Attribute gemessen werden, sondern Attribute von Objekten und diese abstrakte Darstellung von Entitäten hilft dabei die menschliche Sichtweise zu konkretisieren. „*Measurement is a process whose definition is far from clear-cut*“⁴ [FP97]. Folgendes Zitat stimmt oft sehr genau, weil die Mittel und der Prozess zur Durchführung einer Messung oft subjektiven Ansichten unterworfen sind.

Daraus ergibt sich, dass Messen immer kontextabhängig ist und vier weitere Grundsatzfragen:

- Sind Messvorgänge mit potenzieller oder tatsächlich eintretender Fehlerquelle generell falsch?
- Oder welche Fehlerquote ist akzeptabel?
- Welcher Maßstab ist für die Messung korrekt?
- Wie kann man die Resultate analysieren, beurteilen und vergleichen?

² <http://www.raffiniert.ch/galilei.html>

³ [FP97], Seite 5

⁴ [FP97], Seite 5

Diese Probleme können bei jeder Messung auftreten und sind daher stets im Hinterkopf zu bewahren. Um also falsche Ergebnisse zu vermeiden sollte bei jedem Messvorgang

genau darauf geachtet werden. Jede Messung wird also mit einem bestimmten Ziel und Nutzen ausgeführt. Also sollte nicht nur bei problematischen Projekten gemessen werden, sondern generell bei jedem Projekt. DeMarco's Regel besagt: „*You can neither predict nor control what you cannot measure*“⁵ [FP97]. Dies führt zu den dreistufigen Basisaktivitäten für die Messen wichtig ist. Zuerst ist es durch Messen möglich Dinge zu **verstehen**, da Zusammenhänge sichtbar gemacht werden, wodurch im nächsten Schritt **Kontrolle** erlangt wird, dass die Zusammenhänge steuern kann und Voraussagen erlaubt. Folglich können dadurch **Verbesserungen** gemacht werden, die auch präventiven Charakter haben.

1.3 Übertragung auf Software-Engineering

Software-Engineering ist im Vergleich zu anderen Wissenschaften noch sehr jung und Messen ist momentan oft noch ein Luxus und besonders die Übertragung von bewährten Messprinzipien auf sie gestaltet sich noch als sehr schwierig, beispielsweise schon das Festlegen von messbaren Zielen. Gilb's Principle of fuzzy targets lautet: „*projects without clear goals will not achieve their goals clearly*“⁶ [FP97]. Dies beschreibt einen ähnlichen verwandten Sachverhalt. Jedoch sollen unregelmäßige, widersprüchliche und unvollständige Messungen vermieden werden. Als nachteilig gestaltet sich das oft noch geringe Verständnis von Softwareattributen, wie z.B. Komplexität, sowie die Tatsache, dass es kaum gute Tools zum Messen gibt.

Dennoch gehen andere Ingenieurwissenschaften mit gutem Beispiel voran, doch weil „Software Measurement“ aus vielen Teilgebieten besteht gestaltet sich hier eine Messung oft als sehr komplex und dadurch besonders schwierig. Um hier erfolgreich Messungen durchführen zu können müssen die Ziele also spezifisch und klar formuliert sein und beim Messen sollte „mutig“ vorgegangen werden. Mutig bedeutet in diesem Zusammenhang zum Beispiel, dass nur weil etwas bis jetzt noch nicht gemessen wurde es deswegen noch lange nicht unmöglich oder unnötig ist.

Viele Entwicklungsprojekte scheitern schließlich auch am Verstehen und Quantifizieren von Komponentenkosten, bei der Bewertung und Prognostizierung der Softwarequalität oder schlicht und einfach wegen der Benutzung von neuen Technologien, die zuvor weder hinsichtlich Effektivität oder Effizienz überprüft wurden.

⁵ [FP97], Seite 14

⁶ [FP97], Seite 10

2 Theorie des Messens

2.1 Representational Theory of Measurement

Folgende Fragen sind bei klassischen nichtkomplexen Messungen oft leicht zu beantworten:

- Wie viel muss man über Attribute wissen um sie messen zu können?
- Woher wissen wir ob wir das richtige Attribut gemessen haben?
- Welche sinnvollen Aussagen kann man nun treffen?
- Welche aussagekräftigen Operationen können nun durchgeführt werden, bzw. sind sinnvoll?

Bei Softwaremessungen sind diese Grundsatzfragen aber äußerst schwer zu beantworten. Dennoch sollten sie aber beantwortet werden, um eine Messung zum Beispiel zu überprüfen oder zu evaluieren. Um diese Fragen also beantworten zu können benötigt man eine Messtheorie.

Eine solche formale Messtheorie ist die „*Representational Theory of Measurement*“. Diese Theorie basiert auf der „*Representation Condition*“. „*The Representation Condition asserts that a measurement mapping M must map entities into numbers and empirical relations into numerical relations in such a way that the empirical relations preserve and are preserved by the numerical relations*“⁷ [FP97]. Die Repräsentationsbedingung bedeutet also, dass bei einer Messung die empirischen Relationen der Entitäten der Realität in den numerischen Relationen erhalten bleiben und das die numerischen Relationen die empirischen Relationen bewahren.

Somit formalisiert sie die Welt, durch die Übertragung von Objektattributen in Symbole oder Zahlen, das bedeutet den Übergang von der empirischen Welt in die numerische Welt. Die Daten aus den Messungen stellen Attribute von Objekten dar, mit denen dann Manipulationen durchgeführt werden können.

Trotzdem werden Zusammenhänge nicht sofort verstanden wenn sie Nummern zugewiesen bekommen, sondern erst wenn man Vergleiche mit anderen Messungen machen kann. Die Theorie kann also nicht nur helfen, wie gemessen wird, sondern auch wie die Daten zu analysieren und darstellen sind und wie man die Messergebnisse wieder mit der Ursprungsfrage z.B. nach Qualität oder Produktivität zurückverknüpft.

⁷ [FP97], Seite 31

2.2 Empirische Relationen

Beginn einer Messung ist das Feststellen von empirischen Relationen in der Realität, da diese im Zuge des Messvorganges in die numerische Welt transferiert werden sollen.

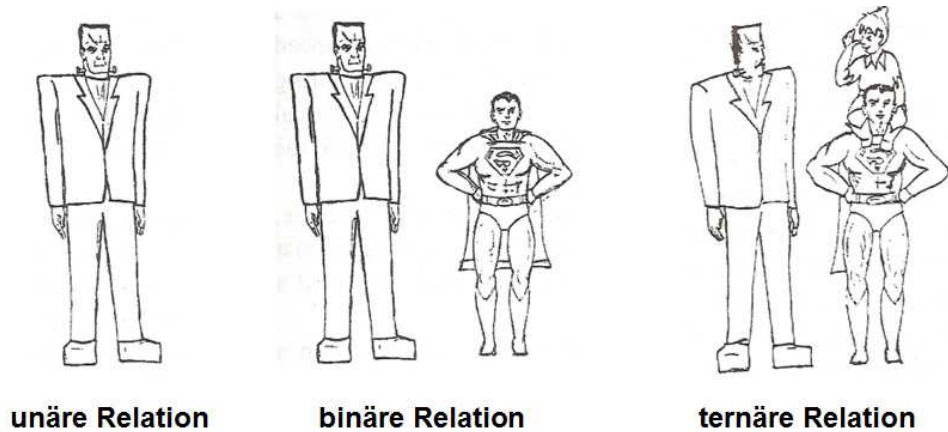


Abbildung 1: empirische Relationen [FP97]⁸

Bei einer unären Relation ist nur eine Entität Gegenstand der Betrachtung, Person X könnte man hier also als „groß“ beschreiben. In einer binären Relation ist hier z.B. Person X „größer als“ Person Y oder in einer ternären Relation sind Y und Z aufeinander „größer als“ Person X. Somit handelt es sich beim Zusammenhang „größer als“ um eine empirische Relation für Körpergrößen, die sobald sie erfasst wurde nun in ein numerisches Relationssystem übertragen werden kann.⁹

Doch oft sind keine objektiven Aussagen möglich, da es unterschiedliche Vorstellungen von einem Attribut geben kann, wie z.B. der Qualität von Wein. Hier können zuerst nur subjektive Aussagen getroffen werden, da jeder Mensch eine andere Vorstellung von einem exzellenten Wein hat. Jene subjektiven Meinungen können aber gesammelt werden und so durch Bildung einer Grundübereinstimmung die Basis für zukünftige empirische Relationen sein und so formales Messen in Zukunft möglich machen.¹⁰

⁸ [FP97], Seite 25

⁹ [FP97], Seite 25

¹⁰ [FP97], Seite 29f

2.3 Mapping

Ist nun eine empirische Relation festgestellt möchte man sie schließlich auch quantifizieren und bewerten können. Doch zuvor muss die Relation noch in eine numerische umgewandelt werden. Dies geschieht durch das sogenannte „Mapping“. Mapping überträgt den gewählten Ausschnitt der Realität in die mathematische Welt. Dort kann er als Zahl, Integer, Symbol etc. dargestellt werden, somit erfolgt gleichzeitig eine Abstraktion und Spezifikation. Nun kann man die Definition von Fenton und Pfleger (Seite 1) erweitern, so dass es schließlich *„measurement is a mapping from the empirical world to the formal, relational world. Consequently, a measure is the number*

or symbol assigned to an entity by this mapping in order to characterize an attribute.“ [FP97] heißt.¹¹

Beim Mapping gibt es eine standardisierte 5-schrittige Vorgehensweise:¹²

- Identifizierung der Attribute der Entitäten in der Realität
- Identifizierung der empirischen Relationen derer
- Identifizierung der korrespondierenden numerischen Relationen
- Definition des Mapping von der Realität in die mathematische Welt
- Überprüfung ob die numerischen Relationen die empirischen wiedergeben und umgekehrt (mithilfe der representation condition)

Das Mapping muss immer eindeutig sein. Versteht man zum Beispiel unter $M(X)$ die Messung, eine Übertragung von einem empirischen Relationssystem in ein numerisches Relationssystem, des Attributes X , so muss für jedes X , das größer als Y ist, $M(X) > M(Y)$ sein, bzw. darf $M(X)$ nur größer als $M(Y)$ sein wenn $X > Y$ ist!

Jede Messung, die die Darstellungsbedingung erfüllt ist folglich eine korrekte Messung und je besser das System der empirischen Relationen, desto häufiger gibt es gültige Messungen. Doch die Kehrseite der Medaille ist hier dann, dass nun bei jeder Messung mehr beachtet werden muss.

¹¹ [FP97], Seite 28

¹² [FP97], Seite 33

2.4 Indirektes Messen

Doch mit der Methode des Mappings kann man nur direkte Messungen durchführen. Manche Attribute von Entitäten können aber aus unterschiedlichen Gründen nicht direkt erfasst werden, weil beispielsweise noch keine adäquaten Tools zum Messen des gewünschten Attributs vorhanden sind. Dies gelingt schließlich nur durch indirektes Messen. Durch Referenz auf andere, bereits messbare Attribute kann man dann auf „Umwegen“ das gewünschte Attribut messen. Bei der Softwareentwicklung gibt es folgende Beispiele für indirekte Messungen:

- Produktivität eines Programmierers = Lines of Code (LOC) / Mannmonate
- Defektdichte eines Moduls = Anzahl der Defekte / Modulgröße
- Fehlererkennungseffizienz = Anzahl der erkannten Fehler / Gesamtanzahl der Fehler

2.5 Prognosen

Mit Messungen wird natürlich nicht nur der Projektstatus etc. festgestellt, sondern sollen beispielsweise auf Grundlage von bestimmten Messwerten Prognosen für die Zukunft getroffen werden können. Eine Prognose besteht aus 3 Schritten, die am Beispiel der Berechnung der Kosten einer Fahrt von München nach Berlin erläutert werden.

- Problem in einem mathematischen Modell darstellen (Kosten = Strecke in km * Preis pro gefahrenem km in €)
- Parameter für die Prognose definieren (hier: aktueller Spritpreis)
- Prozedur zur Interpretation entwickeln (Vergleiche mit anderen Pkws)

Bei 500km und einem Spritpreis von 1,50 € pro Liter mit dem man 10km fahren kann, kommt man hier auf 75€. Da es sich bei diesem Ergebnis nur um eine Prognose handelt und Stau, Umleitungen, überteuerte Autobahntankstellen oder das permanente Rasen des Fahrers den Wert verändern können, ist es kein absoluter Wert, sondern ein Näherungswert im Bereich der Parameter.¹³

2.6 Skalenniveau

Weil es bei gleichen empirischen Relationen oft mehrere Arten des Mapping gibt, gestaltet sich die Wahl der richtigen Skala zur Darstellung dann oft als sehr schwierig. Grundsätzlich sollte man folgende Fragen bezüglich der unterschiedlichen Mappingarten klären:

¹³ [FP97], Seite 42ff

- Wie bestimmt man, wann ein numerisches Relationssystem einem Anderen vorzuziehen ist?
- Woher wissen wir, ob ein bestimmtes empirisches Relationssystem eine Repräsentation in einem numerischen Relationssystem hat?
- Was machen wir, wenn wir unterschiedliche mögliche Repräsentationen (in mehreren Skalen) im gleichen numerischen Relationssystem haben?

Unter dem 2. Punkt versteht man das *Repräsentationsproblem*, also ob die empirischen Relationen korrekt von den numerischen aufgezeigt werden und unter dem 3. Punkt das *Eindeutigkeitsproblem*. Hier ist es schwierig die richtige Auswahl zu treffen beziehungsweise drängt sich die Frage auf wann welche Darstellungsform vorzuziehen. Grundsätzlich kann man sagen, je mächtiger das empirische Relationssystem, desto eingeschränkter sind die Repräsentationsmöglichkeiten, wodurch das Skalenniveau (Messskala) immer differenzierter wird.

2.6.1 Nominalskala

Bei der Nominalskala unterscheidet man nur zwischen verschiedenen Klassen in die dann eingeteilt wird, diese sind in keiner Weise geordnet. Jede Zuweisung auf eine Zahl oder ein Symbol der Klassen ist eine akzeptable Messung und jede 1-zu-1 Abbildung von M zu M' erlaubt. So eine Abbildung von einer Messung auf eine Andere nennt man Transformation. Nur bei der Nominalskala sind alle Transformationen erlaubt, solange es sich um eine 1-zu-1 Transformation handelt. Das bedeutet, dass jede Messung, die die Entitäten gleich einteilt, zulässig ist. Dafür sind nur = und \neq Operationen möglich und es ist möglich den Modus zu bilden. Ein klassisches Beispiel hierfür ist die Einteilung einer Gruppe von Individuen in Mann und Frau.

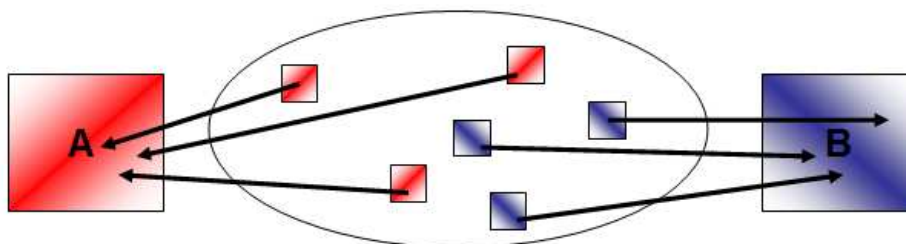


Abbildung 4: Nominalskala

2.6.2 Ordinalskala

Die Ordinalskala ist prinzipiell eine Nominalskala in geordneter Reihenfolge. Alle monotonen Abbildungen von M zu M' mit $M(x) > M(y) \implies M'(x) > M'(y)$ sind als Transformationen erlaubt. Die Transformationen sind schon etwas eingeschränkter als bei der Nominalskala, da alle zulässigen Mappingarten den Gesetzen der Monotonie folgen müssen. Nun sind zusätzlich zu den Operationen der Nominalskala noch $<$ und $>$ Operationen erlaubt, um die Reihenfolge aufzeigen zu können. Somit erfolgt nicht nur eine einfache Einteilung in Klassen, sondern auch eine Anordnung der Klassen in einer Reihenfolge. Zusätzlich kann der Median gebildet werden. Die Unterteilung der Leistung von Schülern in Noten (von „sehr gut“ bis „ungenügend“) ist ein Beispiel für

die Ordinalskala.¹⁴

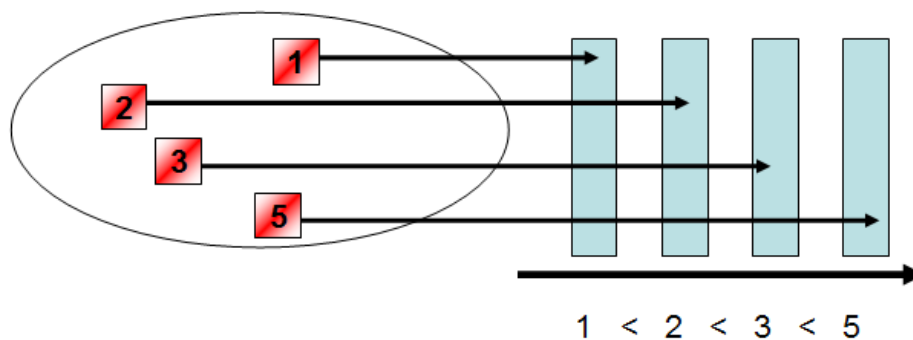


Abbildung 5: Ordinalskala

2.6.3 Intervallskala

Die Intervallskala, die zum Teil auch als Kardinalskala bezeichnet wird, ist schließlich differenzierter als die Ordinalskala, da sie zusätzlich noch die Intervallgröße zwischen den Klassen angibt. Dennoch zeigt sie bloß Differenzen auf, noch kein Verhältnis. Hier sind nur sogenannte Affintransformationen erlaubt, das heißt nach dem Muster $M' = aM + b$ ($a > 0$). Bei der Intervallskala sind außerdem noch $+$ und $-$ Operationen erlaubt und ein Beispiel ist die Darstellung der Temperatur in Fahrenheit oder in Celsius, sowie eine Zeitskala. Ebenso lässt sich das arithmetische Mittel bilden.¹⁵

¹⁴ [FP97], Seite 48f, 53

¹⁵ [FP97], Seite 49ff, 53

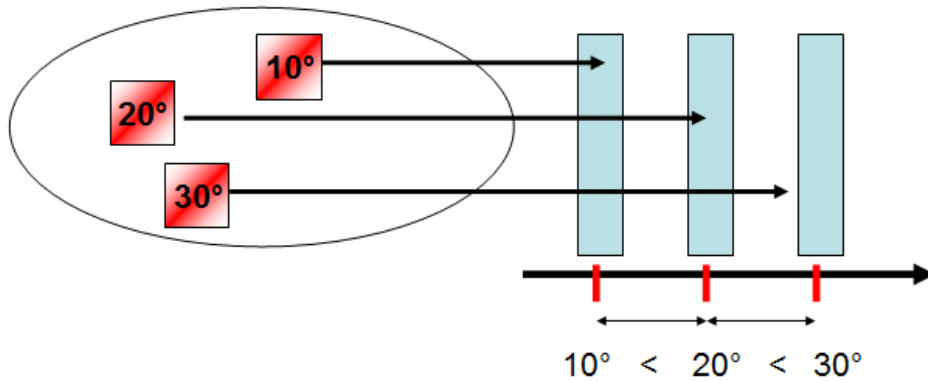


Abbildung 6: Intervallskala

2.6.4 Verhältnisskala

Eine um einen Nullpunkt erweiterte Intervallskala bezeichnet man als Verhältnisskala. Dieser Punkt fungiert als Bezugsobjekt der Verhältnisse, bei dem die Abbildung beginnt.

Bei dieser Skala sind alle arithmetischen Operationen erlaubt und es kann das geometrische Mittel gebildet werden. Möglich sind nur Verhältnistransformationen, also $M' = aM$ ($a > 0$). Die Verhältnisskala benutzt man bei Temperaturmessungen in Kelvin, bei Körpergrößen oder bei Zeitintervallen. Durch den festgelegten Nullpunkt können nun neben Intervallgrößen Relationen zwischen den Messwerten hergestellt werden.¹⁶

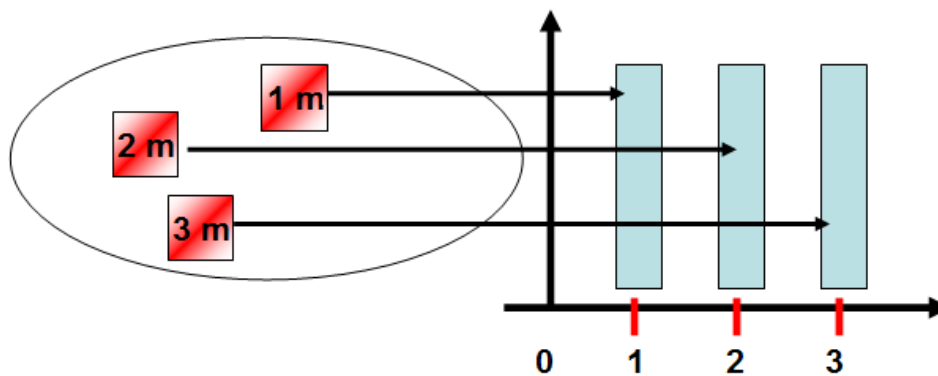


Abbildung 7: Verhältnisskala

¹⁶ [FP97], Seite 51f, 53

2.6.5 Absolute Skala

Die absolute Skala kommt nur durch Zählen der Anzahl der Elemente von Entitäten zu einem Messergebnis, deswegen hat das Attribut immer die Form „Häufigkeit des Ereignisses X in der Entität“. Aus diesem Grund gibt es nur eine korrekte mögliche Messabbildung, nämlich die der Form $M' = M$. Zudem sind alle arithmetischen Operationen erlaubt. Ein Beispiel für diese Skala ist der LOC (Seite 6) als Messergebnis für die Codezeilen in einem Programm.¹⁷

2.7 Simpson Paradoxon

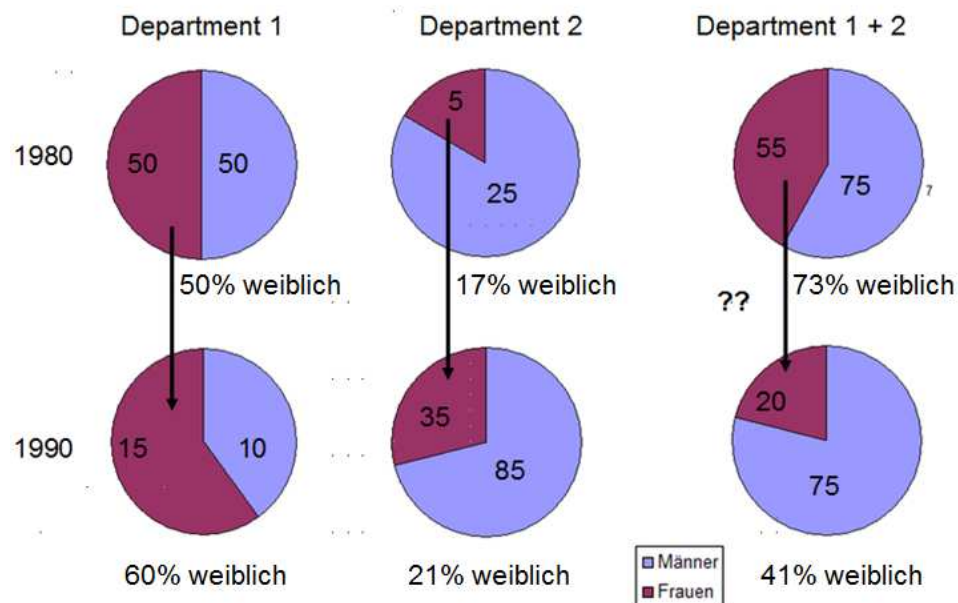


Abbildung 4: Simpson Paradoxon

¹⁷ [FP97], Seite 52f

Im Beispiel aus Abb. 4 nimmt separat betrachtet der weibliche Anteil jeder Abteilung von 1980 bis 1990 zu. Zusammen betrachtet nehmen sie im gleichen Zeitraum aber deutlich ab! Aus diesem Grund kommt es hier zu einem Paradoxon. Das Simpson Paradoxon zeigt uns wie wichtig es ist auf die Betrachtungsweise einer Messung zu achten. Prozentual gesehen nehmen zwar die Frauen in beiden Abteilungen zu, aber in der ersten Abteilung, also der Abteilung in der die meisten Frauen arbeiten, gab es so viele Entlassungen, dass der Gesamtwert trotzdem deutlich sinkt! Beim Simpson Paradoxon handelt es sich um eines der erstaunlichsten Messeffekte. Es zeigt uns, dass man bei einer Messung immer auf alle relevanten Einflüsse auf ein Attribut achten sollte um zu keinem falschen Ergebnis wie im oberen Beispiel zu kommen, da hier beide Abteilungen beweisen können, dass ihre Frauenquote steigt, obwohl sie insgesamt abnimmt.¹⁸

3 Zusammenfassung

Erst die unterschiedlichen Messskalen machen eine korrekte Darstellung unserer Messungen möglich. Durch sie gelingt es die aus den empirischen Relationssystemen gewonnen numerischen Relationen sinngemäß zu präsentieren. Jedoch liegt letztendlich die Entscheidung für eine bestimmte Messskala auch immer beim Urheber der Messung beziehungsweise des Mappings. Somit sollte also der Zweck und Nutzen einer Messung immer im Vordergrund stehen. Andernfalls kann man sie nicht für Prognosen oder beim Software-Engineering zum Beispiel zur Bestimmung des Projektstatus oder der Kosten benutzen. Dabei spielt es keine Rolle, ob die Messung direkt oder indirekt vorgenommen wurde. Besonders wichtig ist das es sich bei einer Messung nach Fenton und Pfleeger um einen Prozess handelt, der für Verstehen, Kontrolle und Verbesserung unserer Umwelt oder in der SE für unser Softwareprojekt zuständig ist und dem deswegen ein so hoher Stellenwert in jedem Bereich der Naturwissenschaften und sogar im Alltag des Menschen zugerechnet wird.

¹⁸ [Zu98], Seite 46f

Literaturverzeichnis

- [FP97] Fenton, N.E., Pfleeger, S.L (1997). *Software Metrics – A Rigorous & Practical Approach - 2nd Edition*. Boston. PWS Publishing Company
- [Zu98] Zuse, Horst (1998), *A Framework of Software Measurement*. Berlin. Walter de Gruyter & Co.