der code ist alles, was struktur ist...

der code ist alles, was für eine software die welt ist. ein code wird durch virtuelle strukturen bestimmt und nicht durch symbolische orientierungen oder fiktionale erwartungen. einzig die varianz aller möglichen strukturen legt fest, was der strukturbare fall ist und was sich zueinander verhält.

in codierungen sind ordnungen und gleichfalls unordnungen als strukturelle korrelationen zu finden. das strukturierbare zeichnet sich durch interne regelmässigkeiten wie wiederholungen und symmetrien aus oder durch unbestimmte dispositionen, die schwer einzuordnen und zu dechiffrieren sind.

die komplexität eines codes hängt von seinen potentiellen verknüpfungen ab. alle zustände sind in der lage, untereinander variable verbindungen einzugehen, sowohl zwischen gleichartigen als auch divergierenden elementen. im einfachsten fall ist der code wie beim password oder einer chiffrierten botschaft jedoch nur ein gefüge von fest miteinander verbundenen zeichen.

für operationen liegen redundante elemente vor, d.h. mögliche, aber nicht zwingend notwendige zeichen oder zeichenkombinationen. sie stabilisieren als zusätzliche details vorliegende lücken sowie unschärfen eines programmcodes und garantieren weitere anschlussmöglichkeiten. dadurch ist für software-entwicklungen ein permanenter aus- und umbau von anwendungen gesichert.

ein code lässt sich als eine anordnung unterscheidbarer und abzählbarer zeichen auf algorithmen zurückführen. sie bestimmen als funktionen seine komplexität und legen mit ihrem funktionsumfang insgesamt fest, welche strukturen erlaubt und welche nicht erlaubt sind. dieserart entstehen auf codes beruhende und aus codes sich generierende programmatiken.

das codierbare ist die abgrenzung einer durch regeln festgelegten zeichensequenz von der gesamtheit aller anderen möglichen, rein zufällig oder permutativ entstandenen zeichenkonstellationen. damit ist eine codierung immer konträr auf den zustand der entropie, d.h. der ungeordneten ungewissheit bezogen. bei einer

völligen gleichverteilung von zeichen ohne erkennbare ordnung wäre kein zustand bestimmt und jede struktur ungewiss.

algorithmen, die codes produzieren, sind verfahrensweisen mit logischen schritten. sie operieren als übergeordnetes system, in dem es keine übergeordneten regeln gibt. alle syntaktischen optionen basieren auf verbindliche austauschbeziehungen, die intern oder systemübergreifend über schnittstellen erfolgen.

die organisation logischer operationen erfolgt ab einer gewissen datenmenge durch substrukturen. in ihnen wird entschieden, welche konnexionen und interaktionen realisierbar sind. durch modifikationen sowie kombinationen werden dabei zumeist solche strukturen angestrebt, die einen zuwachs an komplexität oder eine relative stabiltät versprechen.

im gegensatz zur energie gibt es keinen erhaltungssatz für codierungen. codes können sich in programmen vervielfältigen, wachsen und schrumpfen oder völlig verloren gehen. was erhalten werden soll, muss extern gespeichert werden, da archivierungen in prozessen bei begrenzten kapazitäten temporär angelegt sind.

komplexe ordnungen entstehen wie choatische zustände in sequenziellen abläufen, insofern sie sich selbst organisieren und evolutionär weiterentwickeln. sie zeichnen sich durch eine hohe skalierbarkeit und flexible standards aus. als performative codes können sie erfolg haben, überflüssig werden und manchmal destruktiv zur selbstzerstörung des systems oder sogar von hardware führen.

strukturen in codes sind stets selbstbezüglich, und dies auch auf die gefahr eines regressus ad infinitum, der schleife ohne ende. ihre operationen beziehen sich so lange rekursiv auf sich selbst oder auf andere interne prozesse, bis eine abbruchbedingung erfüllt ist. so bleiben codes als ganzes gesehen tautologisch.

die reine selbstbezüglichkeit ist das organisationsprinzip für komplexe operationen, insofern sie sich nicht auf externe zustände beziehen, sondern nur an den möglichkeiten von inklusiven verknüpfungen orientiert sind. die strukturelle entwicklung ist damit immer operational geschlossen und kann sich nur auf weitere strukturen beziehen. für seine genese in einem computerprogramm benötigt ein code keine information, die nicht bereits in ihm selbst angelegt wäre.

