# der code ist alles, was struktur ist

der code ist alles, was für eine software der fall ist. ein code wird durch virtuelle strukturen bestimmt und nicht durch vorliegende implementierungen oder hardwarekomponenten. erst die varianz aller möglichen programmierungen legt fest, was strukturierbar ist.

in codierungen sind ordnungen und gleichfalls unordnungen als permutative korrelationen zu finden. das strukturierbare zeichnet sich durch interne regelmässigkeiten wie wiederholungen und symmetrien aus oder wuchert in änigmatischen zeichen-agglomerationen.

die komplexität eines codes hängt von seinen potentiellen verknüpfungen ab. strukturelle zustände gehen untereinander variable verbindungen ein, respektive wechselbeziehung zwischen gleichartigen als auch divergierenden datenmengen. im einfachsten fall ist der code wie beim password ein gefüge von beliebig angeordneten zeichen.

codierungen brauchen redundante elemente, d.h. mögliche, aber nicht zwingend notwendige zeichenkombinationen. sie stabilisieren als zusätzliche details die lücken sowie unschärfen eines programmcodes und garantieren weitere anschlüsse. für software-entwicklungen ist mit ihnen 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 fest, welche strukturen relevant und welche nicht relevant sind. so entstehen auf codes beruhende und aus codes sich generierende programmatiken.

das codierbare ist die abgrenzung einer operational generierten zeichensequenz von der gesamtheit aller anderen möglichen, rein zufällig oder permutativ entstandenen konstellationen. damit ist eine codierung immer konträr auf den zustand der entropie, der ungeordneten gewissheit 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, das über keine übergeordneten regeln verfügt. alle syntaktischen optionen basieren auf verbindliche beziehungen von zeichen, welche intern oder systemübergreifend in schnittstellen erfolgen.

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

komplexe ordnungen entstehen wie chaotische zustände in sequenziellen abläufen, wo sie sich selbst organisieren und evolutionär weiterentwickeln. performative codes mit einer hohen skalierbarkeit und flexiblen standards können erfolg haben, überflüssig werden oder auch destruktiv zur selbstzerstörung des systems und sogar der hardware führen.

strukturen in codes sind generell 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 immer tautologische angelegenheiten.

selbstbezüglichkeit ist das organisationsprinzip für komplexe operationen, insofern sie sich nicht auf externe zustände beziehen, sondern an den potentialen von inklusiven verknüpfungen orientiert sind. die strukturelle entwicklung bleibt operational geschlossen, so sie sich auf weitere strukturelle entwicklungen beziehen kann. für seine genese in einem computerprogramm benötigt ein code keine operationale konfiguration, die nicht bereits in ihm angelegt wäre.

 