Ur *Mellan Himmel och Jord*, KTH, 1995.

Trollkarlens hatt

Tillräckligt utvecklad teknologi kan ofta likna magi. Trollkarlens hatt, mjölnarens kvarn och sekreterarens dator är alla lika häpnadsväckande för den oinvigde. Men för utövaren är de artskilda. Kvarnens ingenjörer begriper hur de olika delarna fungerar och passar ihop. Den som använder magi däremot riskerar alltid oväntade och oönskade resultat eftersom en trollkarl inte är herre över sin konst.

På detta sätt kan man med fog klassa datorer som trolltyg. Dataexperten är vår tids häxmästare som frambesvärjer fantastiska och förfärande ting utan att behärska dem. Datorsystem kontrollerar nu allt från telefonväxlar till flygplan och kärnkraftverk trots att ingen, inte ens deras skapare, kan redogöra för varför och under vilka förutsättningar de fungerar.

Modell och verklighet

Ingenjörskonsten är högt utvecklad inom många områden och dess adelsmärke är att konstruktion inleds med omfattande planering. När man ska bygga en kvarn börjar bara amatören med att spika vid stranden. Yrkesmannen gör istället först ritningar och modeller med vilka det är möjligt att räkna ut viktiga egenskaper hos det som ännu inte är byggt. Om till exempel ett kugghjul blir för klent upptäcks detta redan i modellen och man kan ändra konstruktionen.

Vi informationsingenjörer däremot arbetar i bästa fall med ofullständiga och tvetydiga ritningar. Oftast skruvar vi snabbt ihop hela systemet, och försöker kompensera bristen på planering med att utsätta det för så många tester vi orkar. Men tidiga flygturer kan lätt sluta i "Gripen-gropen". Funktionsfel kan inte förutsägas och när de yttrar sig i den lakoniska dödsrosslingen "system error" blir det ett styvt jobb att spåra orsaken och ett sisyfosarbete att åtgärda den, eftersom varje förändring kan rubba andra delar av systemet.

I datorernas barndom var minnet mycket litet och programmen följaktligen små och få. Programmerarna satte en ära i att hålla hela system i huvudet och dödade "lusar", programfel, genom att tänka djupt och länge. Idag innehåller den enklaste persondator miljontals

programinstruktioner, och ändå är försök till sådana intellektuella maratonlopp legio när systemen ska avlusas. Anledningen är att det saknas effektiva alternativ. Vi vet inte hur ritningar och modeller ska se ut för att vara till riktig hjälp. När vi kan göra sådana för rymdraketer, järnvägsbroar och pacemakers, varför går de inte att göra för datorsystem?

Tanke och materia

Man säger ofta att informationsteknologin är en ung vetenskap som inte ännu hunnit resultera i en utvecklad ingenjörskonst. Detta kan dock knappast vara hela sanningen. Datorn som teknisk uppfinning är ungefär jämnårig med både atombomben och jetplanet och äldre än rymdraketen.

En rimligare förklaring är att informationsteknologin behandlar immateriella konstruktioner som inte låter sig beskrivas med välkända mekaniska egenskaper som vridmoment och friktion. När datorerna arbetar är deras fysiska attribut oviktiga, det intressanta är deras beteende. Men exakt vad menas med "beteende" och i vilka termer kan det göras begripligt? Man vill gärna tänka på det materiellt: knappar trycks in, lampor tänds, signaler åker genom ledningar. Eller man kan se det antropomorfiskt: datorn tror, maskinen vill, systemet surar. Men de materiella liknelserna är för begränsade och antropomorfin för vag för att vara ritningsunderlag.

I vanliga ingenjörsdiscipliner sätter materien en gräns för storlek och uppbyggnad; datorsystemens "beteenden" begränsas bara av programmerarnas uppfinningsrikedom. Så länge vi inte ens har en terminologi för att beskriva dem kan vi inte hoppas på att bygga dem ingenjörsmässigt.

Form och funktion

Till viss del är materiella analogier användbara. Flödet genom ett datanät kan behandlas med samma matematiska metoder som trafiken på en motorväg. Många dimensioneringsproblem, som optimering av exekveringstid och minnesåtgång kan angripas på liknande sätt.

I andra fall söker forskningen nya vägar. Så kräver exempelvis AXE-telefonernas stjärnor och fyrkanter ett kvalitativt snarare än kvantitativt resonemang, där det viktiga är hur funktionerna kan användas och kombineras och inte hur hårt knapparna kan tryckas in. Man vill kunna uttrycka egenskaper som "om abonnenten har beställt vidarekoppling kommer inkommande samtal att skickas vidare" utan att lämna utrymme för olika tolkningar. Därmed blir telefonens funktioner tillgängliga för

en logisk analys, som kan avslöja att telefonen blir obrukbar om den vidarekopplas till sig själv eller att telefonväckning inte fungerar om man har vidarekoppling. Funktionsbeskrivningen är alltså en slags modell som används för att förutsäga beteendet.

Forskningsproblemen här är att definiera logiska språk och metoder där resonemangen blir enkla, helst så enkla att till och med en dator kan utföra delar av dem. En bärande idé för att hantera stora beskrivningar är att sammanföra funktionerna i grupper som kan analyseras separat. Men för att de ska vara ritningar i ingenjörskonstens mening måste det också gå att bygga system efter dem. Detta är särskilt svårt för distribuerade system, alltså system som består av flera samverkande delar på olika platser. Systemets form och uppdelning i komponenter bildar en struktur medan dess funktioner bildar en annan. En funktion kan beröra flera komponenter och en komponent kan i sin tur bidra till flera funktioner.

Först de senaste åren har resultaten från denna forskning börjat visa sig. I modern systemutveckling insisterar man på en begriplig struktur för både form och funktion. Nya språk tvingar programmerarna att ange hur olika programdelar kan kombineras för att undvika att de sätts ihop på fel sätt. Inför säkerhetskritiska tillämpningar i kärnkraftsverk krävs en logisk analys som bevisar att beteendet är acceptabelt. Men övergången från kaotisk till välordnad produktion är inte smärtfri. Den inskränker ju frihet och uttryckssätt för alla våra dataäventyrare och programmeringskonstnärer.

Datorerna har kommit för att stanna men deras svartkonst kan vi klara oss utan. Som så ofta förut bygger ingenjören hus där magikern brutit mark. Och det som nu är trollkarlens hatt blir mjölnarens kvarn för våra barn.

Joachim Parrow, Professor i Distribuerade System, KTH