SYSTEMIC ASPECTS OF CHOOSING BASIC ARCHITECTURAL PATTERNS IN SOFT-WARE SYSTEMS DESIGN

Dr. Gerhard Chroust, Prof. Emer.,
Johannes Kepler University,
Linz, Austria
Erwin Schoitsch
Austrian Research Centers – ARC,
Vienna, Austria

When designing a complex software-intensive system it is unavoidable to make some a-priori basic assumptions about its architecture. We define 7 Fundamental Properties of software Intensive systems: Complexity, Flexibility, Dependability, Survivability, Time consumption, Resources utilization, Ease of use. We introduce so-called Basic Architectural Alternatives (Architectural Alternatives) as a means to guide these decisions and to understand their effects. These alternatives are classified according to five fundamental dimensions: Enactment time, Location, Granularity, Control, Human/computer task distribution. For these Architectural Alternatives we describe 4 Fundamental Properties: Elasticity, Uniformity, Reversibility and Method Applicability. For each dimension we describe some typical, real examples of such an alternative together with its Fundamental Properties.
Finally we discuss synergetic or contradictive effects of Architectural Alternatives when combined. We show how Architectural Alternatives balance one another or enforce one another, depending on their basic behavior. We hope that this analysis together with the specific examples and their key properties provide some insight for students, novices and seasoned designers and guides them during the early phases of system design.

SISTEMSKI VIDIKI IZBIRE TEMELJNIH VZORCEV ZA SESTAVLJANJE DELOV V CELOTO, KADAR KONSTRUIRAMO SISTEME RAČUNALNIŠKIH PROGRAMOV

Dr. Gerhard Chroust, Prof. Emer.,
Johannes Kepler University,
Linz, Austria
Erwin Schoitsch
Austrian Research Centers – ARC,
Vienna, Austria

Kadar konstruiramo kompleksen sistem z veliko računalniških programov, je nujno izoblikovati nekaj vnaprejšnjih predpostavk, kako ga bomo sestavili. Opredeljujemo sedem temeljnih lastnosti za take sisteme: kompleksnost, prilagodljivost, zanesljivost, sposobnost preživeti, porabo časa, uporabo virov in lahko uporabo. Uvajamo tako imenovane temeljne alternative pri sestavljanju programov (arhitekturne alternative) kot pripomočke, da usmerjajo te odločitve in dosežejo razumevanje njihovih učinkov. Te alternative razporejamo v razrede po petih osnovnih vidikih: čas izvedbe, lokacijo, granularnost, obvladovanje in delitev dela med človekom in računalnikom.
Za te arhitekturne alternative opišemo štiri temeljne lastnosti: prožnost, enoličnost, ponovljivost in uporabnost metode. Za vsak vidik opišemo nekaj tipičnih stvarnih primerov take alternative skupaj z njenimi temeljnimi lastnostmi.
Ob koncu razpravljamo o sinergijskih ali med seboj nasprotnih učinkih arhitekturnih alternativ, kateri se pojavi, če jih kombiniramo. Prikažemo, kako te arhitekturne alternative spravljajo druga drugo v ravnovesja ali krepijo, kar je odvisno od njihovega obnašanja. Upamo, da taka analiza skupaj s posebnimi primeri in njihovimi ključnimi lastnostmi omogoča vpogled študentom, novincem in občasnim konstruktorjem/oblikovalcem programov ter jih vodi skozi zgodnje faze oblikovanja računalniških programskih sistemov.