



INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

Modelování a simulace

12. lekce

Michal Janošek

Programovací prostředky pro modelování a simulaci

- * Rozbor možností programovacích prostředků
- * Simulační programovací jazyky
- * Klasifikace simulačních jazyků (A, AT, T)
- * Jazyky s elementárními prvky
- * Kombinovaná simulace

Simulační program

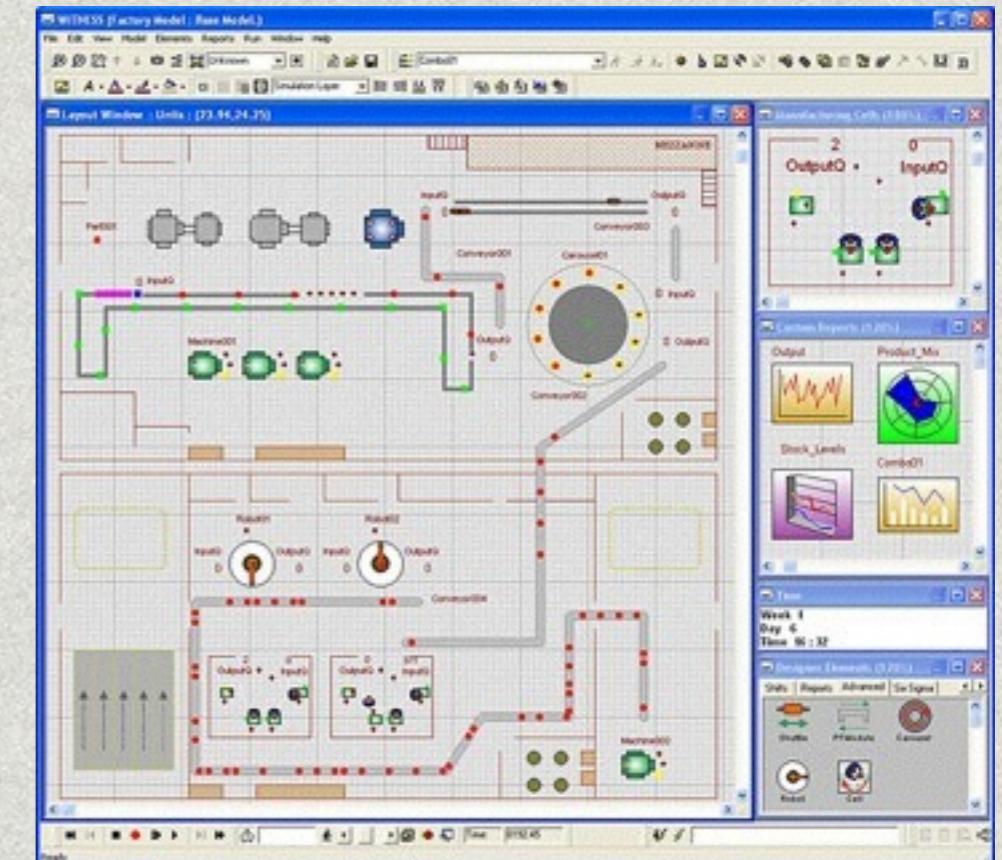
- * program řídící výpočet
 - * program v programovacím jazyku
 - * program ve strojovém kódu
- * simulační program není simulátor
 - * tím se stane počítač, kterým je tímto programem řízen
- * simulační programovací prostředky
 - * simulační jazyky, a další..
- * simulační programy jsou často velmi složité
 - * výpočet musí být analogií složitého děje

Časté způsoby implementace simulačních programů

1. použití předem připraveného „univerzálního“ modelu, který je řízen daty tak, že některá z nich jeho běh směrují různými cestami;
2. použití programovacích jazyků tak, že texty v nich sestavené a popisující více méně čitelně požadavky na to, co se má počítat, jsou překládány do odpovídajících programů ve strojovém kódu;
3. použití objektově orientovaných programovacích prostředků jakožto základu pro definice adekvátních problémově orientovaných programovacích prostředků.

1. Předem připravený univerzální model

- * rozmanité aplikační obory
 - * výroba počítačů, strojírenství, služby
 - * dispečeři energetických sítí, letištní pracovníci, pracovníci ve zdravotnictví
- * reagují na kombinovaná alfanumerická a grafická vstupní data
 - * velká rozmanitost dat
- * programy Taylor, Witness
 - * <http://www.lanner.com/en/witness.cfm>



Witness

Superb Visual Communication

Superb Visual Communication



Superb Visual Communication

Superb Visual Communication

<http://www.youtube.com/watch?v=NSHea6UZYuo>

2. Programovací jazyky popisující požadavky

- * požadavky jsou překládány do strojového kódu

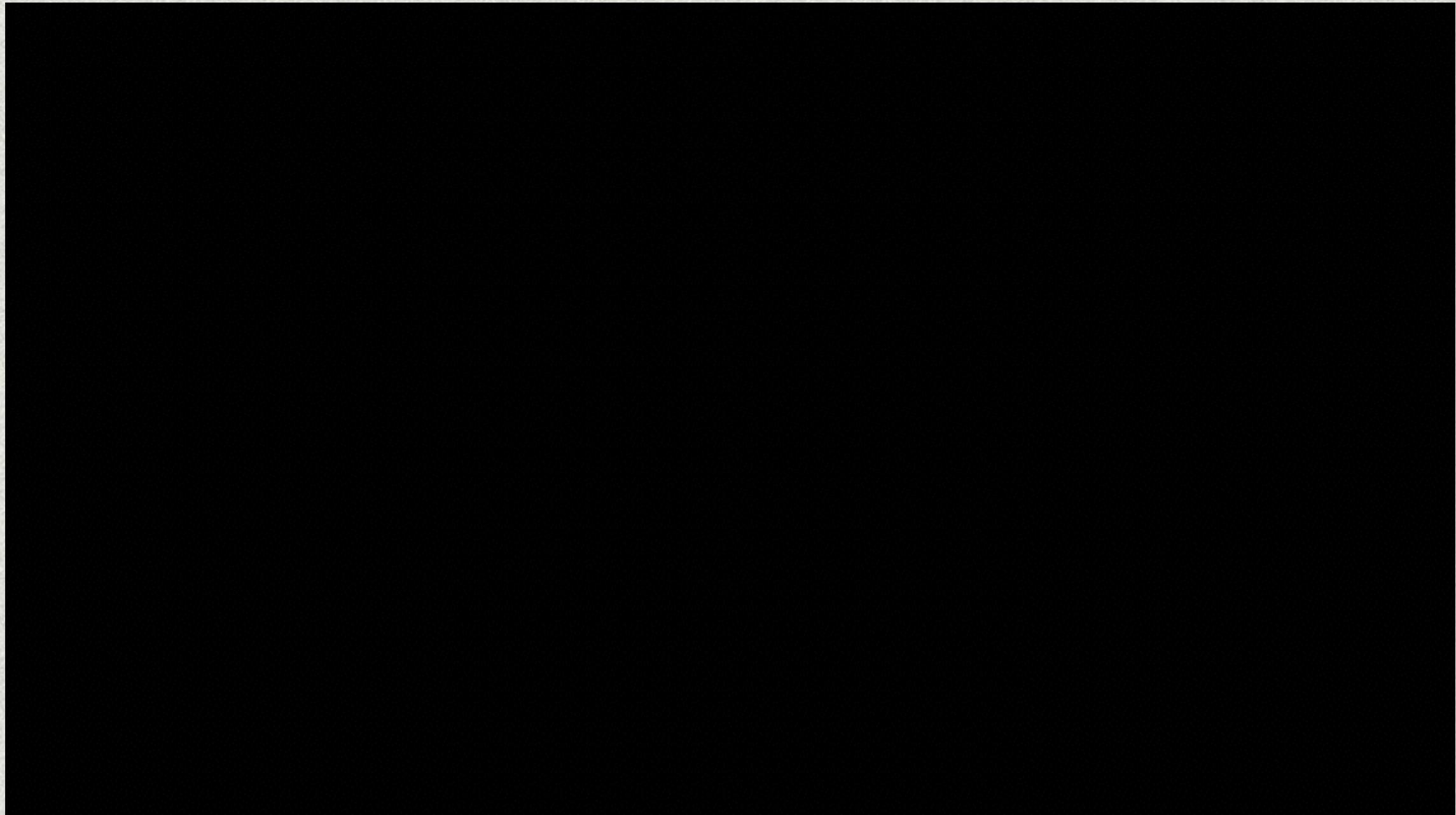
- * **simulační programovací jazyky (stovky)**

- * Arena
- * ExtendSim
- * NetLogo
- * ACSL
- * EcosimPro
- * Flexsim
- * GPSS

```
S1 STORAGE 10
S2 STORAGE 12
GENERATE 10,,0,1000
SEIZE Z1
ADVANCE 6,3
RELEASE Z1
ENTER S1
ADVANCE 40,15
LEAVE S1
SEIZE Z2
ADVANCE 5,2
RELEASE Z2
ENTER S2
ADVANCE 50,20
LEAVE S2
SEIZE Z3
ADVANCE 4,2
RELEASE Z3
TERMINATE
GENERATE 5,3,30,600
```

GPSS

Flexsim



<http://www.youtube.com/watch?v=5UcB5xtr4gg>

3. Objektově orientované prostředky

- * OOP
- * Simula (1967)
- * SmallTalk (1980)
- * Java
- * C#

```
process class generator2;
begin loop:activate new vyrobek2;
    hold(normal(5,3,U));
    if počet2 lt 600 then go to loop
end;
transaction class vyrobek1;
begin počet1:=počet1+1;
    seize(Z1); hold(normal(6,3,U));
    release(Z1);
    enter(S1); hold(normal(40,15,U));
    leave(S1);
    seize(Z2); hold(normal(5,2,U));
    release(Z2);
    enter(S2); hold(normal(50,20,U));
    leave(S2);
    seize(Z3); hold(normal(4,2,U));
    release(Z3)
end;
```

SIMULA

Simulační programovací jazyky

- * též simulační jazyky
- * pomoc při sestavování simulačních programů
- * simulovaný systém byl již popsán v jiném jazyku než programovacím
 - * výrobní provoz, nervová soustava pacienta
 - * odborná čeština, angličtina
- * adaptovat jazyk profese na programovací jazyk
 - * stanovit jeho přesná syntaktická a sémantická pravidla
 - * odstranit nebezpečí nejasností, které přináší vždy přirozený jazyk
 - * umožnit strojový překlad textů v takto upraveném jazyku přímo do simulačních programů.

Sémantické třídy simulačních jazyků

- * mnoho specializovaných simulačních jazyků
- * jazyky vhodný pro popis systémů patřících do jisté sémantické třídy
 - * pro jiná systémy nevhodný
- * lékař těžko popíše fyziologii nějakého orgánu v jazyku ekonomu, atd.
- * Např. v nukleární terapii nádorů je terminologie, které rozumějí odborníci v této profesi; příslušné termíny by museli vysvětlit např. jiným lékařům, s nimiž by v dané problematice přišli do styku, ale na základě obecné odborné terminologie je takové vysvětlení možné, zatímco definovat termíny nukleární nádorové terapie způsobem přijatelným dejme tomu ekonomovi nebo odborníku v hutnictví oceli je takřka nemožný.
- * základní klasifikace simulačních jazyků

Základní klasifikace simulačních jazyků

- * v systémech existují transakce a aktivity (perm. prvky)
- * **systémy (jazyky) A** - aktivity
 - * bez transakcí, zanedbáváme všechny strukturní změny
- * **systémy (jazyky) AT** - aktivity, transakce
 - * obsahují prvky obou kategorií
- * **systémy (jazyky) T** - transakce
 - * proměnné v čase, bez aktivit
- * Pozn. jazyk AT může popsat systém A i T

Jazyky s elementárními prvky

- * životní pravidla - analogie s algoritmy
- * 3 ideje
- * snaha použít k popisu životních pravidel zvyklostí (včetně konkrétních syntaktických pravidel) známých a ustálených v konvenčních algoritmických jazycích
 - * Fortran, Basic, Java
- * životní pravidla (celé popisy simulovaného systému, tedy celé texty v simulačním jazyku) mohly automaticky překládat do textů v nějakém konvenčním algoritmickém jazyku
- * algoritmické struktury nějakého rozšířeného programovacího jazyka by se doplnily vhodnými podprogramy pro generování a likvidování transakcí a pro plánovací příkazy

Jazyky s elementárními prvky

- * Životní pravidla mají tvar

```
ADVANCE 50,20
LEAVE S2
SEIZE Z3
ADVANCE 4,2
RELEASE Z3
TERMINATE
```

- * A₁, P₁, A₂, P₂, ..., A_n, P_n, A_{n+1},
- * kde A_i jsou úseky, které lze vyjádřit pomocí prostředků konvenčních algoritmických jazyků,
- * P_i jsou plánovací příkazy, dejme tomu tvaru „čekej po dobu T_i“ nebo „čekej, až bude splněna podmínka B_i“ .
- * Definujeme v systému n + 1 tříd E₁, E₂, ..., E_n, E_{n+1} jakýchsi fiktivních prvků
- * **elementární prvky** – které jsme původně v systému nepozorovali.
- * Každý z těchto elementárních prvků má jeden referenční atribut R a s třídou E_i jsou spojena pravidla A_i, za nimiž následuje ještě doplněk D_i, který má tvar „generuj prvek třídy E_{i+1}, okopíruj na jeho atribut R současnou hodnotu atributu R a aktivuj tento prvek za čas T_i (resp. až bude splněna podmínka B_i).“
- * elementární prvky lze převést na podprogramy
- * nepoužívají se (SIMSCRIPT I a II)

Kombinovaná simulace

<http://www.fit.vutbr.cz/~peringer/SIMLIB/doc/html-cz/node44.html.en>

- * kombinovaná simulace se chápe jako simulace systému, který má podstatné vlastnosti jak systémů spojitéch, tak i systémů diskrétních.
- * v kombinovaném systému tedy existují prvky, jejichž některé atributy se mohou měnit (alespoň někdy) spojite, a dále v něm existují diskrétní události, včetně vzniku a zániku transakcí.
- * Vyskytuje-li se v nějakém systému spojité a diskrétní změny, lze právem očekávat, že se v něm tyto změny dostanou i do nějaké vzájemné závislosti, a tak jazyky pro kombinovanou simulaci musí být vybaveny prostředky pro popis těchto závislostí.
 - * Diskrétní událost způsobí diskrétní změnu hodnoty atributu, která se jinak mění spojite.
 - * Spojitě se měnící atribut dosáhne jistého stavu, což způsobí diskrétní událost, která se může projevit na zcela jiných vlastnostech systému než na daném atributu.
- * Jazyk GASP IV a GASP V, pomocí třídy SIMULATION v jazyku SIMULA

Použité obrázky

- * <http://dynamicfuture.cz/obrazky/uvodka.jpg>