

Demonstrační simulační projekt EM Banka

Návod k implementaci projektu v prostředí aplikace SIMPROCESS

Obsah

1.	ÚVOD	2
2.	ZALOŽENÍ NOVÉHO PROJEKTU	2
3.	ENTITY, ZDROJE, ATRIBUTY ENTIT, ATRIBUTY MODELU	2
3.1.	DEFINICE ENTITY (ENTITY) ZAKAZNIK A JEJÍCH ATRIBUTŮ (ATTRIBUTE)	2
3.2.	DEFINICE ZDROJŮ (RESOURCE) BANKOMATA, BANKOMATB, BANKOMATC	2
3.3.	DEFINICE ATRIBUTŮ MODELU (MODEL ATTRIBUTES)	3
4.	NÁVRH NEJVYŠŠÍ ÚROVNĚ V HIERARCHII PROCESŮ A AKTIVIT	3
4.1.	ANALÝZA.....	3
4.2.	PROCESS PRICHODYZAKAZNIKU.....	3
4.3.	PROCESS OBSLUHAZAKAZNIKU	4
4.4.	ACTIVITY ODCHODYZAKAZNIKU	4
4.5.	SPOJENÍ VLOŽENÝCH PROCESŮ A AKTIVIT POMOCÍ BENT CONNECTOR	4
4.6.	VIZUÁLNÍ KONTROLA	4
5.	KONSTRUKCE PODPROCESŮ PROCESU PRICHODYZAKAZNIKU	5
5.1.	ANALÝZA.....	5
5.2.	PŘESUN DO NIŽŠÍ ÚROVNĚ V HIERARCHII PROCESŮ	5
5.3.	ACTIVITY GENERATE - GENEROVÁNÍ ENTIT ZAKAZNIK	5
5.4.	ACTIVITY BRANCH, ASSIGN A MERGE.....	6
5.5.	VIZUÁLNÍ KONTROLA	8
6.	KONSTRUKCE PODPROCESŮ PROCESU OBSLUHAZAKAZNIKU	8
6.1.	ANALÝZA.....	8
6.2.	POJMENOVÁNÍ IMPLEMENTOVANÉ VARIANTY	9
6.3.	VYUŽITÍ AKTIVIT ASSIGN, BRANCH, DELAY A MERGE PRO SIMULACI OBSLUHY ZÁKAZNÍKŮ	9
6.4.	VIZUÁLNÍ KONTROLA	10
6.5.	POPIS FUNKČNÍHO PRINCIPU.....	10
6.6.	DEFINICE UŽIVATELSKÉ FUNKCE PRO VOLBU BANKOMATU	10
6.7.	POUŽITÍ FUNKCE VOLBANKOMATUABC.....	11
6.8.	VĚTVENÍ NA ZÁKLADĚ HODNOTY ATRIBUTU ZVOLENYBANKOMAT ENTITY ZAKAZNIK.....	11
6.9.	DEFINICE UŽIVATELSKÉ FUNKCE PRO VÝPOČET DOBY OBSLUHY	12
6.10.	POUŽITÍ FUNKCE DELKAOBSLUHYNABANKOMATU	12
7.	DISPOSE ACTIVITY ODCHODYZAKAZNIKU	13
8.	PRVNÍ SPUŠTĚNÍ SIMULAČNÍHO BĚHU	13
9.	SBĚR A ZOBRAZOVÁNÍ STATISTICKÝCH DAT	14
9.1.	VESTAVĚNÝ NÁSTROJ PRO SBĚR STATISTICKÝCH DAT - NASTAVENÍ	15
9.2.	VESTAVĚNÝ NÁSTROJ PRO SBĚR STATISTICKÝCH DAT - PŘIDÁNÍ SLEDOVANÝCH ATRIBUTŮ	15
9.3.	ZOBRAZENÍ STANDARDNÍHO STATISTICKÉHO PŘEHLEDU	16
9.4.	SLEDOVÁNÍ VLASTNÍCH STATISTICKÝCH UKAZATELŮ	16
10.	NÁPISY A DYNAMICKÝ TEXT (BACKGROUND TEXT / DYNAMIC LABEL)	18
10.1.	VLOŽENÍ STATICKÉHO TEXTU	19
10.2.	POUŽITÍ DYNAMICKÉHO TEXTU.....	19
11.	ZÁVĚREČNÉ TESTOVÁNÍ	20

1. Úvod

Tento manuál krok za krokem popisuje postup implementace demonstračního simulačního projektu EM Banka v prostředí aplikace SIMPROCESS dle daného zadání, resp. varianty se třemi bankomaty a oddělenými frontami.

Koncepce manuálu předpokládá jeho postupné studium spolu se současným vykonáváním popsaných kroků v prostředí aplikace SIMPROCESS.

2. Založení nového projektu

- Spustíte aplikaci SIMPROCESS
- Z menu zadejte příkaz File / New
- Z menu zadejte příkaz File / Save As ... a uložte projekt pod zvoleným jménem do předem připravené složky

3. Entity, zdroje, atributy entit, atributy modelu

V rámci implementace daného simulačního modelu budeme pracovat s entitou Zakaznik, jež bude při obsluze využívat jeden ze tří dostupných bankomatů. Abychom s uvedenou entitou a zdroji mohli v simulovaném modelu pracovat, je třeba nejprve zavést jejich definice a specifikovat jejich atributy níže popsaným způsobem.

3.1. Definice entity (ENTITY) Zakaznik a jejích atributů (ATTRIBUTE)

Při definici entity Zakaznik postupujte následovně:

- Z hlavního menu zadejte příkaz Define / Entities ...
- V dialogovém okně stiskněte tlačítko Add ...
- Textové pole Name vyplňte názvem deklarované entity, tj. Zakaznik
- Změňte vzhled ikony - v roletce Icon zvolte např. optimističtější vypadající BlueDot
- Potvrďte obě dialogová okna (tlačítka OK a Close)

Nyní pro právě definovanou entitu Zakaznik zavedeme definice jejích atributů. Význam těchto atributů bude vysvětlen později.

- Z hlavního menu zadejte příkaz Define / Attributes / Entities ...
- V dialogovém okně stiskněte tlačítko Add ...
- Textové pole Name vyplňte názvem KonecCekaniNaObsluhu
- Z roletky Mode zvolte položku Real
- Pole Value vyplňte hodnotou 0.0
- Potvrďte dialog OK
- Stejným postupem přidejte další atribut, který pojmenujte ZacatekCekaniNaObsluhu
- Stejným postupem přidejte další atribut, který pojmenujte ZvolenyBankomat
- Stejným postupem přidejte atribut TypTransakce, tentokrát však v roletce Mode zvolte String a Value ponechte na 0
- Zavřete dialogové okno tlačítkem Close

3.2. Definice zdrojů (RESOURCE) BankomatA, BankomatB, BankomatC

Vzhledem k tomu, že budeme implementovat variantu zadání 3 bankomaty s oddělenými frontami, je třeba nadefinovat 3 samostatné zdroje, představující jednotlivé bankomaty, každý v počtu 1 jednotka (pokud bychom však implementovali variantu 3 bankomaty se společnou frontou, stačilo by nadefinovat jediný zdroj Bankomat v počtu 3 jednotek).

Při definici zdrojů postupujte takto:

- Z hlavního menu zadejte příkaz Define / Resources ...
- V dialogu stiskněte tlačítko Add ...
- Textové pole Name vyplňte názvem BankomatA
- V poli Units ponechte výchozí hodnotu, tj. 1
- Potvrďte dialogové okno stiskem tlačítka OK
- Stejným postupem definujte zdroje BankomatB a BankomatC
- Zavřete dialogové okno tlačítkem Close

3.3. Definice atributů modelu (MODEL ATTRIBUTES)

Nyní zavedeme globální atributy modelu, jejichž význam bude rovněž popsán později:

- Z hlavního menu zadejte příkaz Define / Attributes / Model ...
- V dialogovém okně stiskněte tlačítko Add ...
- Textové pole Name vyplňte názvem CelkDobaCekani
- Z roletky Mode zvolte položku Real
- Pole Value vyplňte hodnotou 0.0
- Potvrďte dialogové okno OK
- Stejným postupem přidejte další 2 atributy, které pojmenujte PocetCekNad3Min a PocetCekPod3Min - z roletky Mode však zvolte Integer a pole Value ponechte vyplněné celočíselnou hodnotou 0
- Uzavřete dialogové okno stiskem tlačítka Close

4. Návrh nejvyšší úrovně v hierarchii procesů a aktivit

4.1. Analýza

Po provedení prvotní analýzy implementovaného modelu dospějeme ke zjištění, že problém se rozpadá na tři základní procesy, které je třeba simulovat:


- Příchody zákazníků
- Obsluha zákazníků na bankomatech
- Odchody obsloužených zákazníků

Složitější procesy Příchody zákazníků a Obsluha zákazníků na bankomatech se budou v rámci hierarchie budovaného modelu skládat z dílčích podprocesů a aktivit, proto na nejvyšší úrovni v hierarchickém modelu stavěném v aplikaci SIMPROCESS budou zastoupeny prostřednictvím objektů PROCESS, zatímco simulovaný jev Odchody obsloužených zákazníků bude zastoupen elementárním objektem ACTIVITY, neboť jeho jediným úkolem bude rušení entit Zákazník poté, co vystoupí z procesu Obsluha zákazníků a nebude se rozpadat na další dílčí podprocesy či aktivity.

4.2. PROCESS PrichodyZakazniku

- Klikněte na tlačítko PROCESS  na paletě




- Klikněte uvnitř okna modelu - tím objekt PROCESS  Process1 vložíte do projektu - jeho výchozí název zvolí SIMPROCESS automaticky, např. Process1.

- Operací drag-and-drop umístíte ikonu cca do 1/4 zleva v horizontálním směru a na střed ve směru vertikálním
- Klikněte levým tlačítkem na vloženém objektu PROCESS a lokálního menu zvolte položku Properties ...
- Text v políčku Name změňte na PrichodyZakazniku - tím přiřadíte objektu identifikátor, jež můžete využít v uživatelsky definovaných scriptech. Jméno se rovněž zobrazí u objektu namísto původního, automaticky zvoleného jména
- Dialog potvrďte stiskem tlačítka OK
- Klikněte myší na malém symbolu trojúhelníku umístěném na levé straně objektu PROCESS pojmenovaném PrichodyZakazniku a vymažte jej stiskem klávesy Delete na klávesnici. Objekt PrichodyZakazniku představuje počátek "dráhy", po níž se budou pohybovat entity (ty zde budou vznikat, tj. budou zde generovány), proto tento objekt bude obsahovat pouze výstupní bod (symbol trojúhelníku vpravo), zatímco vstupní bod je třeba právě popsáním způsobem zrušit


4.3. **PROCESS ObsluhaZakazniku**

- Zcela analogickým postupem jako v bodě 4.2 vložte do středu plochy modelu PROCESS, který pojmenujte ObsluhaZakazniku (v případě tohoto objektu však již vstupní bod neodstraňujte)

4.4. **ACTIVITY OdchodyZakazniku**

- Klikněte na tlačítko DISPOSE  a kliknutím na ploše modelu a vložte tak tento objekt do projektu
- Klikněte na nově vloženém objektu DISPOSE pravým tlačítkem myši a zvolte položku Properties...
- Text v políčku Name změňte na OdchodyZakazniku a potvrďte dialogové okno stiskem tlačítka OK

4.5. **Spojení vložených procesů a aktivit pomocí BENT CONNECTOR**

- Klikněte na tlačítko BENT CONNECTOR 
- Klikněte na symbolu trojúhelníku vystupujícího z procesu PrichodyZakazniku a následně klikněte na symbolu trojúhelníku vstupujícího do procesu ObsluhaZakazniku
- Analogickým postupem spojte procesy ObsluhaZakazniku a OdchodyZakazniku
- Klikněte pravým tlačítkem myši na první z vytvořených spojovacích čar a z lok. menu zvolte Properties, dále zrušte zaškrtnutí u položky Show Name. Totéž provedte s druhou spojovací čarou

4.6. **Vizuální kontrola**

Po provedení výše popsaných kroků by plocha vámi vytvářeného modelu měla vypadat takto:



5. Konstrukce podprocesů procesu PrichodyZakazniku

V tomto okamžiku předpokládám, že jste si prostudovali, prakticky vyzkoušeli a zvládnuli postupy popsané v kapitole 4. V dalším textu již tedy nebudu podrobně popisovat dílčí kroky (klikněte pravým tlačítkem myši, ...) při operaci umístování komponent na plochu modelu a změnu jejich vlastností (dialogové okno Properties).

5.1. Analýza



Nyní již máme vytvořenou kostru budovaného modelu, přesněji máme v základu hotový návrh nejvyšší úrovně v hierarchii procesů. Aplikace SIMPROCESS je výrobcem deklarována jako hierarchický nástroj pro tvorbu simulačních modelů. Umožňuje nám proto vytvářet modely strukturovaně a postupovat při tvorbě modelu od nejjobecnější úrovně až k přesné definici konkrétních vztahů a událostí.

V této kapitole se zaměříme na již vytvořený objekt - PROCESS PrichodyZakazniku a budeme definovat, co se děje "uvnitř něho", tj. umístíme do něj příslušné aktivity (event. bychom mohli do něj umístit další podprocesy).

5.2. Přesun do nižší úrovně v hierarchii procesů


"Dovnitř" daného procesu, tj. na nižší úroveň v hierarchii procesů se přesunete pomocí poklepání (levým tlačítkem) myši na ikoně příslušného procesu. Pokud tedy poklepete na objektu procesu PrichodyZakazniku, v okně projektu se vám otevře plocha modelu, zobrazující podprocesy a aktivity, z níž se proces PrichodyZakazniku skládá. V našem případě bude tato plocha prozatím prázdná.

Případný návrat na vyšší resp. nejvyšší úroveň v hierarchii procesů provedete stiskem tlačítka

ASCEND  resp. GO TO TOP  v horním panelu. Nyní se tedy prosím přemístěte "dovnitř" procesu PrichodyZakazniku.

Před prováděním dalších kroků kapitoly 5 si prosím prohlédněte obrázek v odstavci 5.5, a příslušné komponenty pak umíst'ujte tak, abyste dosáhli jejich podobného rozmístění na ploše modelu.

5.3. ACTIVITY GENERATE - Generování entit ZAKAZNIK

Pro generování entit v prostředí aplikace SIMPROCESS slouží aktivita GENERATE . Vložte tedy tento objekt na plochu modelu vám již známým postupem.

Nyní prosím proveďte následující kroky:

- Pro právě vloženou aktivitu GENERATE vyvolejte dialog Properties
- Name změňte na GenPrichoduZakazniku
- Dialogové okno ponechte otevřené a pokračujte následujícími kroky

Dále bude třeba definovat, kolik entit Zakaznik bude generováno v kterých časových intervalech a podle kterého statistického rozdělení. Počty příchozích zákazníků v daných intervalech jsou uvedeny v zadání, přičemž předpokládáme, že přichody zákazníků se řídí exponenciálním rozdělením. Zadejme tedy tyto parametry do simulovaného modelu:

- V sekci Shedule oznčte a tlačítkem Remove smažte implicitně vytvořené schéma Periodic1
- V roletce napravo od tlačítka Add zvolte položku Cyclical a stiskněte tlačítko Add


- V nově otevřeném dialogu do pole Shedule zadejte text IntervalyPrichoduZakazniku
- V sekci Sequence of Events ponechte v roletce předvolenou hodnotu Periodic a stiskněte tlačítko Add
- Nyní nadefinujeme předpis pro generování entity Zakaznik v prvním časovém intervalu, tj. od 9:30 do 10:30 hod., proto do políčka Shedule zadejte název Od_0930_Do_1030
- V roletce Interval zvolte položku Exp(10.0) a následně stiskněte tlačítko s třemi tečkami vedle ní
- V čase od 9:30 do 10:30 hod. má do banky přicházet v průměru 95 zákazníků, proto střední hodnota časového odstupu mezi příchody zákazníků bude 37,895 sekund. Tuto hodnotu tedy zadejte do pole Mean. Políčko Stream u prvního intervalu bude obsahovat číslo 1. Dialog potvrďte tlačítkem OK
- Zrušte zaškrtnutí u Generate at start of first interval
- Změňte časovou jednotku Time Unit z Hours na Seconds
- V sekci Event nastavte Duration na 1.0, položku Time Units ponechte na Hours a dialog potvrďte tlačítkem OK
- Dále zopakujte předchozích sedm kroků pro každý z dalších 5 časových intervalů, pro něž je třeba nadefinovat počty příchodů zákazníků. Pro každý interval si musíte spočítat střední hodnotu časového odstupu mezi příchody zákazníků (= 3600 / průměrný počet příchodů). Pro druhý interval použijte Stream 2, pro třetí interval Stream 3 atd. Každý interval pojmenujte příslušným názvem ve stejném stylu, jaký byl použit pro první interval, tj. Od_HHMM_Do_HHMM. Přitom zde s výhodou můžete využít v prvním kroku tlačítko Copy ... namísto Add - pak změníte pouze pojmenování Shedule a hodnotu Interval
- Potvrďte dialogové okno stiskem tlačítka OK

5.4. ACTIVITY BRANCH, ASSIGN a MERGE

Poté, co právě vytvořená GENERATE aktivita GenPrichoduZakazniku při simulačním běhu vygeneruje entitu Zakaznik, bude třeba této entitě přiřadit typ transakce, kterou daný zákazník chce u bankomatu provést. Pro tento účel jsme již dříve nadefinovali atribut TypTransakce, jež ponese každá entita Zakaznik. Nyní tedy musíme zabudovat do simulačního modelu mechanismus, který tento atribut pro každou nově vygenerovanou entitu inicializuje příslušnou hodnotou. Typy transakcí přicházejících v úvahu a jejich relativní četnosti jsou specifikovány v zadání příkladu.

Na příslušná místa na ploše modelu (viz obrázek v odstavci 5.5) nyní umístěte ikonu aktivity

BRANCH , 7x ikonu aktivity ASSIGN  a ikonu aktivity MERGE .

Dále pomocí BENT CONNECTOR  již dříve popsaným způsobem spojte jednotlivé ikony - zde se řiďte rovněž obrázkem v odstavci 5.5. Vzhledem k tomu, že aktivita BRANCH resp. MERGE slouží k větvení resp. slučování symbolických cest, po nichž "putují" entity, bude třeba aktivitu BRANCH resp. MERGE propojit s jednotlivými aktivitami ASSIGN 7x. Rovněž je třeba propojit pomocí BENT CONNECTOR výstup z aktivity MERGE na výstupní bod z procesu PrichodyZakazniku - jedná se o samostatný symbol malého černého trojúhelníčku umístěný zcela vpravo na ploše modelu. Abyste jej spatřili, zřejmě budete muset pomocí horizontální posuvné lišty posunout obsah okna. Všimněte si, že v levé části na ploše

modelu není umístěn vstupní bod - to proto, že jsme jej dříve vymazali (kapitola 4.2), neboť proces PrichodyZakazniku představuje generátor entit a žádné entity do něj nebudou vstupovat.

Následně proveďte tyto kroky:

- Vyvolejte dialog Properties pro BRANCH activity a zrušte zaškrtnutí u Show Name. Dále ověřte, že v sekci Select Branch Type je zvolena varianta Probability (přiřazení hodnoty atributu TypTransakce entity Zakaznik bude probíhat na základě generování náhodných hodnot dle rovnoměrného rozdělení). Nakonec potvrďte dialogové okno stiskem tlačítka OK
- Vyvolejte dialog Properties pro MERGE activity a zrušte zaškrtnutí u Show Name, potvrďte stiskem OK

Dále

- Vyvolejte dialog Properties pro ikonu ASSIGN activity (první shora)
- Do pole Name zadejte Transakce_BM_H
- V sekci Set Attributes v roletce napravo od tlačítka Add zvolte položku Entity.TypTransakce, poté stiskněte tlačítko Add
- Nato se objeví dialogový rámeček, kde hodnotu Operation necháte nastavenou na = a do pole Value zadejte text BM_H. Potvrďte dialog tlačítkem OK
- Výchozí dialog rovněž potvrďte tlačítkem OK
- Výše popsané úkony opakujte postupně pro zbývajících šest aktivit ASSIGN, přičemž jména aktivit (Name) volte dle obrázku v odstavci 5.5. Rovněž hodnotu přiřazovanou ve čtvrtém kroku modifikujte s ohledem na jméno dané aktivity, takže pro druhou aktivitu ASSIGN (počítáno shora) zadáte namísto BM_H hodnotu BM_V atd.

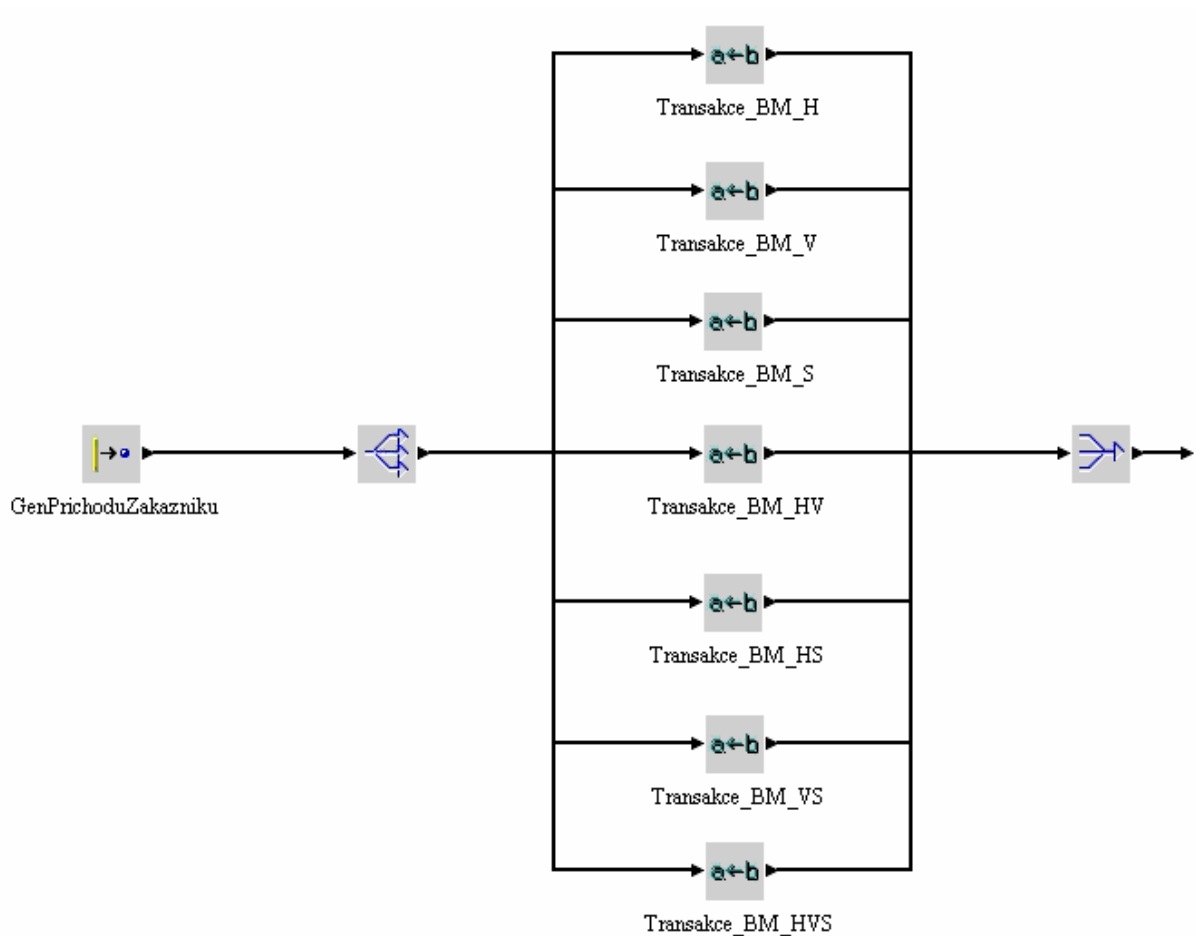
Dále

- Vyvolejte dialog Properties pro spojovací čáru (BENT CONNECTOR), jež propojuje BRANCH activity a první ASSIGN activity (počítáno shora).
- Zrušte zatržítka u Show Name
- V sekci Branch Parameters zvolte variantu If Probability is a zadejte hodnotu 0.4 (protože typ transakce H, tj. výběr hotovosti, se vyskytuje s četností 40%)
- Výše popsaný postup opakujte pro zbývajících šest spojovacích čar, přičemž reálnou hodnotu pravděpodobnosti ve třetím kroku zadejte vždy tak, aby byla v souladu se zadáním příkladu. Součet všech sedmi zadaných hodnot musí přirozeně být 1.

Nakonec již známým způsobem zrušte zobrazování jména (Show Name v Properties) u všech zbývajících spojovacích čar (BENT CONNECTOR), neboť jejich pojmenování zde nemá význam a zbytečně činí model nepřehledným.

5.5. Vizuální kontrola

Po provedení výše popsaných kroků by plocha procesu PrichodyZakazniku měla vypadat takto:



Pokud se vám podařilo úspěšně dokončit všechny kroky kapitoly 5, můžete se vrátit na

nejvyšší úroveň v hierarchii vytvářeného modelu stisknutím tlačítka GO TO TOP



6. Konstrukce podprocesů procesu ObsluhaZakazniku

6.1. Analýza

V této kapitole se zaměříme na výstavbu té části simulačního modelu, v níž probíhá vlastní simulace obsluhy zákazníků. Připomínám, že tento manuál popisuje implementaci varianty zadání 3 bankomaty s oddělenými frontami.

6.2. Pojmenování implementované varianty

Aplikace SIMPROCESS umožňuje vytvořit v jednom projektu více variant simulačního modelu, mezi nimiž se lze přepínat. Definování těchto variant se provádí ve vlastnostech aktivity PROCESS. Námi budovanou variantu simulačního modelu pojmenujeme takto:





- Na nejvyšší úrovni v hierarchii simulačního modelu vyvolejte dialog Properties pro PROCESS ObsluhaZakazniku
- V sekci Alternative Sub-Processes je nyní jediná položka pojmenovaná Alt1. Stiskněte tlačítko Edit... a v nově otevřeném dialogu změňte Name z Alt1 na BM3_F3 (pojmenování odvozeno z implementované varianty, tj. 3 bankomaty a 3 oddělené fronty)
- Potvrďte stisknutím OK

Tímto jsme si tedy pojmenovali aktuálně vytvářenou variantu simulačního modelu. Pokud byste si v budoucnu chtěli rozšířit model o další varianty dle zadání, právě v tomto dialogovém okně můžete pomocí tlačítka Add nebo Copy přidat další položky, tedy další varianty. Poté byste příslušnou položku zvolili a po potvrzení dialogu tlačítkem OK by při spuštění simulačního běhu nebo při editaci podprocesů procesu ObsluhaZakazniku byla platná vámi zvolená varianta.

Nyní tedy potvrďte dialog tlačítkem OK.

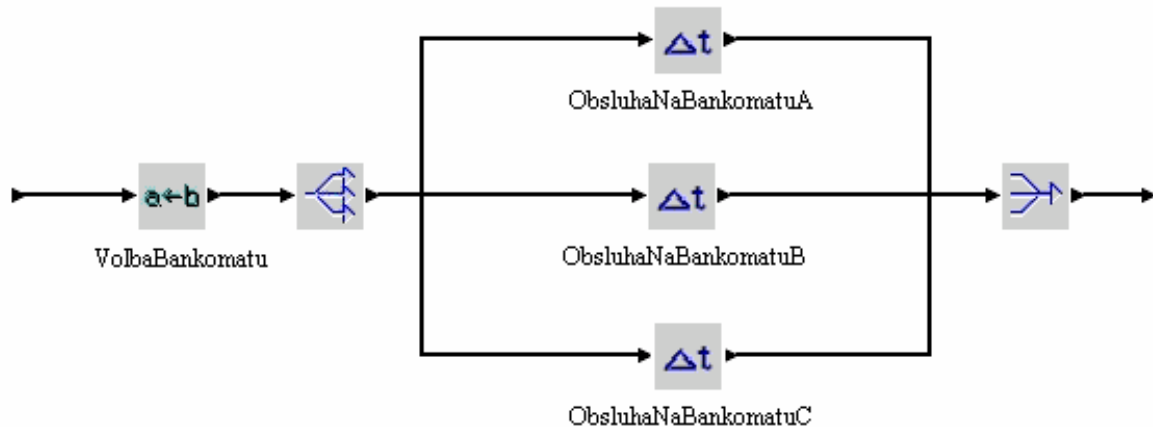
6.3. Využití aktivit ASSIGN, BRANCH, DELAY a MERGE pro simulaci obsluhy zákazníků

Simulování procesu obsluhy zákazníků na bankomatech budeme realizovat prostřednictvím

aktivit ASSIGN , BRANCH , DELAY  a MERGE . Umístěte tedy prosím ikony těchto aktivit na plochu procesu ObsluhaZakazniku, přičemž jejich rozmístění proveďte podle obrázku v kapitole 6.4. Dále pospojíte příslušné ikony pomocí BENT CONNECTOR a přejmenujte příslušné aktivity - rovněž podle zmíněného obrázku. Neopomeňte propojit aktivitu ASSIGN (první zleva) se vstupním bodem podprocesu ObsluhaZakazniku a dále aktivitu MERGE (první zprava) s výstupním bodem.

6.4. Vizuální kontrola

Po provedení kroků popsaných v předchozí kapitole by plocha procesu ObsluhaZakazniku měla vypadat takto:



6.5. Popis funkčního principu

Poté, co ENTITY Zakaznik vstoupí do PROCESS ObsluhaZakazniku, přichází na ASSIGN ACTIVITY VolbaAutomatu. Zde je volána níže popsaná funkce VolbaBankomatuABC, která přiřadí atributu ZvolenyBankomat ENTITY Zakaznik číselnou hodnotu (1-3), představující volbu jednoho ze tří bankomatů. Následně BRANCH ACTIVITY provádí větvení na základě hodnoty tohoto atributu a směřuje ENTITY Zakaznik na příslušnou DELAY ACTIVITY ObsluhaNaBankomatuA/B/C. Na každé z těchto tří DELAY ACTIVITY probíhá simulace vlastní obsluhy zákazníka, tj. realizace příslušné transakce na bankomatu (výběr hotovosti, ...). Výpočet délky doby obsluhy provádí na základě stochastického přístupu další uživatelská funkce DelkaObsluhyNaBankomatu (rovněž popsána níže). Přitom se během simulované obsluhy využívá RESOURCE BankomatA/B/C.

Nyní tedy přistupme k vlastní implementaci popsaného funkčního principu.

6.6. Definice uživatelské funkce pro volbu bankomatu

Ačkoli SIMPROCESS nabízí celou řadu nástrojů a funkcí, které můžeme při výstavbě simulačního modelu přímo využít, v každém netriviálním modelu budeme pravděpodobně potřebovat mít možnost upravit specifickým způsobem jeho chování. A právě k tomuto účelu jsou zde k dispozici uživatelské funkce (FUNCTION) a skripty (EXPRESSION SCRIPT). Nyní si ukážeme, jak nadefinovat funkci, která bude v simulačním modelu implementovat rozhodovací proces, kdy přichází zákazník provádět volbu bankomatu, na němž bude realizovat zamýšlenou transakci. Pokud je v okamžiku příchodu zákazníka některý ze tří bankomatů volný, volí zákazník tento bankomat. V opačném případě vybírá ten, u něhož je nejmenší fronta.

Uživatelskou funkci můžete nadefinovat takto:

- Z hlavního menu zadejte příkaz Define / Functions ...
- Stiskněte tlačítko Add ...
- V nově otevřeném dialogu do políčka Name zadejte název VolbaBankomatuABC
- Stiskněte tlačítko Expression ...
- Do editačního okna vložte text scriptu, který naleznete v následující tabulce (můžete samozřejmě využít kopírování přes schránku)
- Syntaktickou správnost scriptu můžete ověřit pomocí tlačítka Validate
- Je-li vše v pořádku, stiskněte Done
- Stiskněte OK a potom Close

Uživatelská funkce VolbaBankomatuABC

```
{ Deklarace promenných }
FrontaA, FrontaB, FrontaC: INTEGER;

{ Zjisteni poctu entit, které cekaji na obsluhu nebo
  jsou obsluhovany na prislusne DELAY ACTIVITY ObsluhaNaBankomatuA/B/C }
FrontaA:= Sibling("ObsluhaNaBankomatuA").NumberIn;
FrontaB:= Sibling("ObsluhaNaBankomatuB").NumberIn;
FrontaC:= Sibling("ObsluhaNaBankomatuC").NumberIn;

{ Porovnani zjistenyh hodnot a urceni navratove hodnoty funkce }
IF (FrontaA <= FrontaB) AND (FrontaA <= FrontaC)
  RETURN 1;
ELSIF (FrontaB <= FrontaA) AND (FrontaB <= FrontaC)
  RETURN 2;
ELSE RETURN 3;
ENDIF;
```

6.7. Použití funkce VolbaBankomatuABC

Nyní je třeba zajistit, aby funkce VolbaBankomatuABC byla použita ke svému účelu, tj. k nastavení atributu ZvolenyBankomat ENTITY Zakaznik. Postup je následující:

- Vyvolejte dialog Properties pro ASSIGN ACTIVITY VolbaBankomatu
- V sekci Set Attributes v roletce napravo od tlačítka Add zvolte položku Entity.ZvolenyBankomat a stiskněte tlačítko Add
- V nově otevřeném dialogu ponechte položku Operation na = a z roletky Value vyberte hodnotu VolbaBankomatuABC
- Potvrďte oba dialogy tlačítkem OK

6.8. Větvení na základě hodnoty atributu ZvolenyBankomat ENTITY Zakaznik

Nyní je třeba zajistit, aby do modelu námi vložená BRANCH ACTIVITY následující za ASSIGN ACTIVITY VolbaBankomatu správně prováděla větvení na základě přiřazené hodnoty atributu ZvolenyBankomat ENTITY Zakaznik. Postup je následující:

- Vyvolejte dialog Properties pro výše uvedenou BRANCH ACTIVITY
- V sekci Select Branch Type zvolte položku Atribute
- V roletce vedle této položky zvolte položku Entity.ZvolenyBankomat
- Potvrďte dialogové okno tlačítkem OK

Dále pro každý BENT CONNECTOR spojující BRANCH ACTIVITY a DELAY ACTIVITY ObsluhaNaBankomatuA/B/C

- Vyvolejte dialog Properties
- V sekci Branch Parameters u položky If Entity.ZvolenyBankomat is: vyberte položku =
- Do vstupního pole vedle roletky s hodnotou = zadejte hodnotu 1 (pro spojovací čáru vedoucí k DELAY ACTIVITY ObsluhaNaBankomatuA) resp. hodnotu 2 (pro ObsluhaNaBankomatuB) resp. hodnotu 3 (pro ObsluhaNaBankomatuC) - jedná se o návratové hodnoty funkce VolbaBankomatuABC
- Dialogové okno potvrďte OK

6.9. Definice uživatelské funkce pro výpočet doby obsluhy

Další funkcí, kterou budeme potřebovat, je funkce pro výpočet doby obsluhy daného zákazníka na základě typu transakce, kterou chce na bankomatu provést. Střední hodnoty délky obsluhy pro danou transakci jsou uvedeny v zadání příkladu. Zde příslušnou stochastickou hodnotu získáme pomocí zabudované funkce DrawIntegerSample, jíž předáme jako parametr typ statistického rozdělení (exponenciální), střední hodnotu a číslo proudu náhodných čísel. Při definici této funkce postupujte vám již známým způsobem (viz kapitola 6.6) s tím, že modifikujete jméno funkce na DelkaObsluhyNaBankomatu a použijete text scriptu uvedený v následující tabulce.

Uživatelská funkce DelkaObsluhyNaBankomatu

```
DelkaTransakce: INTEGER;

IF      Entity.TypTransakce = "BM_H" DelkaTransakce:=
DrawIntegerSample("Exp(30.0, 7)");
ELSIF  Entity.TypTransakce = "BM_V" DelkaTransakce:=
DrawIntegerSample("Exp(25.0, 8)");
ELSIF  Entity.TypTransakce = "BM_S" DelkaTransakce:=
DrawIntegerSample("Exp(20.0, 9)");
ELSIF  Entity.TypTransakce = "BM_HV" DelkaTransakce:=
DrawIntegerSample("Exp(50.0, 10)");
ELSIF  Entity.TypTransakce = "BM_HS" DelkaTransakce:=
DrawIntegerSample("Exp(45.0, 11)");
ELSIF  Entity.TypTransakce = "BM_VS" DelkaTransakce:=
DrawIntegerSample("Exp(40.0, 12)");
ELSIF  Entity.TypTransakce = "BM_HVS" DelkaTransakce:=
DrawIntegerSample("Exp(65.0, 13)");
ELSE OUTPUT("Chyba - fce DelkaObsluhyNaBankomatu - Neznamy typ
transakce!");
ENDIF;

RETURN DelkaTransakce;
```

6.10. Použití funkce DelkaObsluhyNaBankomatu

Právě nadefinovanou funkci DelkaObsluhyNaBankomatu použijeme při určení délky každé jedné transakce prováděné daným zákazníkem. Tuto operaci simulujeme prostřednictvím tří DELAY ACTIVITY, tj. ObsluhaNaBankomatuA/B/C. Postupně tedy pro každou z těchto tří DELAY ACTIVITY proveďte tyto kroky:

- Pro danou DELAY ACTIVITY vyvolejte dialog Properties
- V sekci Duration změňte hodnotu Units na Seconds, z roletky Value zvolte položku DelkaObsluhyNaBankomatu
- Stiskněte tlačítko Resources ...

- V roletce vpravo vedle tlačítka Add zvolte položku BankomatA (resp. BankomatB resp. BankomatC - podle aktuálně editované DELAY ACTIVITY)
- Stiskněte tlačítko Add
- Všechna dialogová okna potvrďte tlačítkem OK

Tím prozatím ukončíme práci na PROCESS ObsluhaZakazniku, takže se můžeme vrátit na nejvyšší úroveň v hierarchii procesů.

7. DISPOSE ACTIVITY OdchodyZakazniku

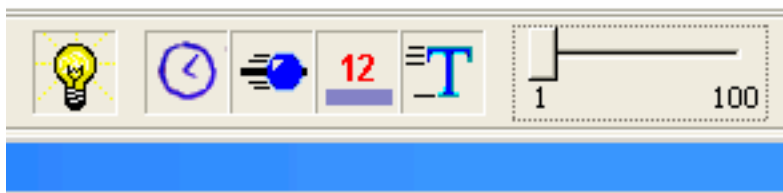
Jediným účelem DISPOSE ACTIVITY OdchodyZakazniku je rušení ENTITY Zakaznik poté, co došlo k ukončení jeho obsluhy. Tuto činnost dokáže DISPOSE ACTIVITY zajistit automaticky, proto z hlediska funkčnosti modelu zde není třeba provádět žádné další modifikace kromě správného propojení této ACTIVITY pomocí BENT CONNECTOR s ostatními částmi modelu, což jsme však již učinili.


8. První spuštění simulačního běhu


Pokud jste správně provedli kroky uvedené v předchozích kapitolách, měl by v tomto okamžiku být náš simulační model v jeho základní verzi hotov, takže můžeme přistoupit k prvnímu spuštění simulačního běhu.

Postupujte prosím takto:

- Z menu zadejte příkaz Simulate / Run Settings ...
- Dialog Run Settings vyplňte podle obrázku níže
- V horním panelu aktivujte všechna tlačítka a nastavte rychlost animace takto



- Spusťte simulaci tlačítkem RUN SIMULATION 
- Tím dojde ke spuštění simulace, což je indikováno textem Simulating ve stavovém pruhu dole, stejně tak jako pořadové číslo simulačního běhu simulační čas. Rovněž byste měli vidět animované (pohybující se) ikony ENTITY Zakaznik (modrá kolečka)
- Simulační běh můžete nechat doběhnout do konce nebo jej můžete kdykoli přerušit

tlačítkem Stop the simulation 

Run Settings

Simulation Period: [mm/dd/yyyy hh:mm:ss:msec:usec:nsec]

Start: 01/01/2003 09:30:00:: ...

End: 01/01/2003 15:30:00:: ...

Warmup Every Replication Number of Replications: 1

Reset Random Number Streams Warmup Length: 0

Reset System Warmup Time Unit: Seconds

Verify Model on Run Simulation Time Unit: Seconds

RMI Host: localhost RMI Port: 1099

Cost Periods...

OK
Cancel
Help

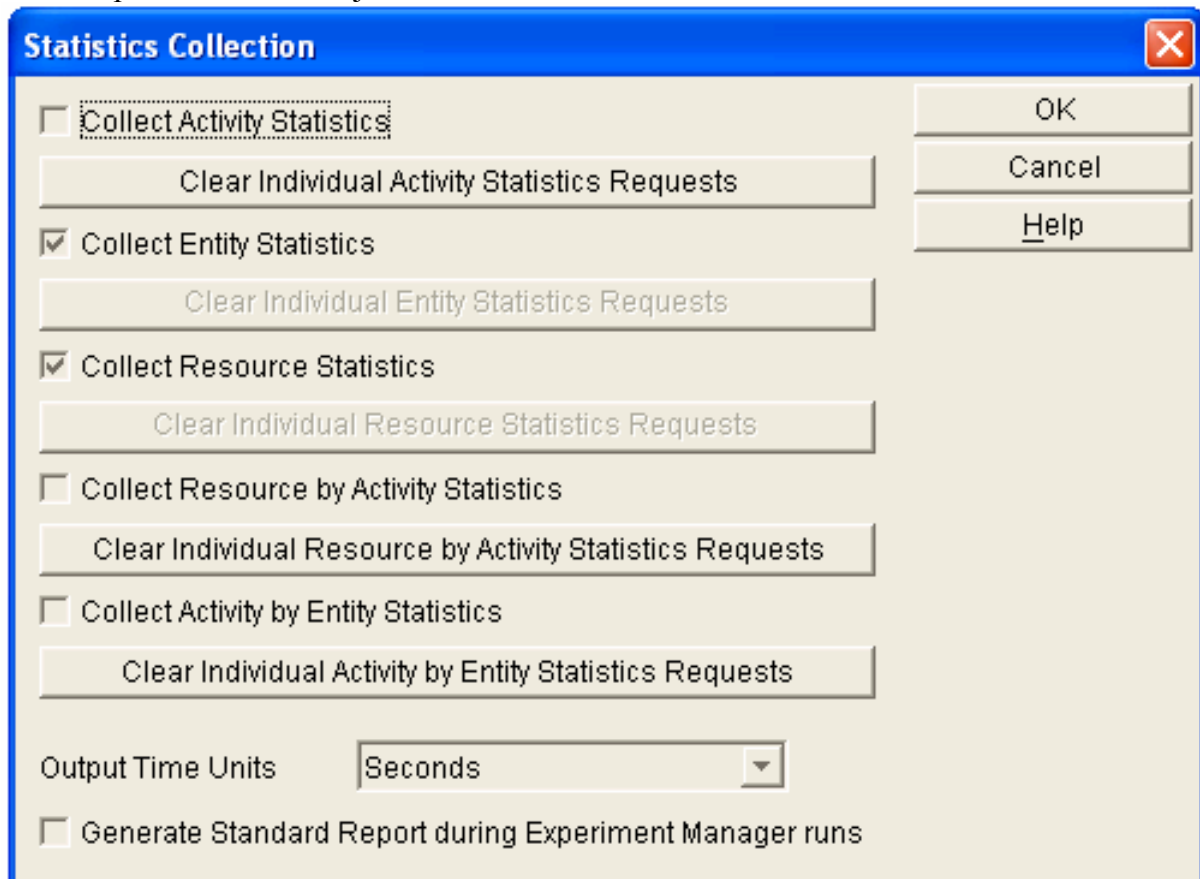
Poznámka: při vytváření vašich dalším simulačních modelů v aplikaci SIMPROCESS vždy důkladně zkontrolujte, zda v dialogu Run Settings máte položku Simulation Time Unit nastavenou na správnou hodnotu, tj. takovou jednotku, kterou jste si zvolili za základní.

9. Sběr a zobrazování statistických dat

Konečným cílem výstavby simulačního modelu je zpravidla vyhodnocení statistických výstupů, které by daná aplikace měla produkovat během a/nebo po ukončení simulačního běhu.

9.1. Vestavěný nástroj pro sběr statistických dat - nastavení

Konfiguraci vestavěného nástroje pro sběr statistických dat, tedy určení kategorií dat, které má tento nástroj během simulačního běhu sledovat a vyhodnocovat, se provede po zadání příkazu z menu Report / Define Global Statistic Collection. Vyvolané dialogové okno nastavte prosím dle následujícího vzoru:



9.2. Vestavěný nástroj pro sběr statistických dat - přidání sledovaných atributů

Aplikace SIMPROCESS vám umožňuje určit skladbu výsledného, automaticky generovaného statistického přehledu. Nyní si ukážeme na příkladech, jak přidat do statistického přehledu některé položky.

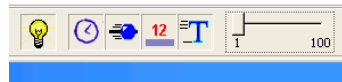
Na počátku výstavby našeho simulačního modelu jsme nadefinovali některé atributy modelu, mj. to byly atributy PocetCekNad3Min a PocetCekPod3Min. Chcete-li přidat výpis výsledných hodnot těchto atributů do statistického přehledu, postupujte takto:

- Z menu zadejte příkaz Define / Attributes / Model ...
- Zvolte položku PocetCekNad3Min
- Stiskněte tlačítko Edit ...
- V sekci Statistic Types nastavte volbu Observation Based
- V sekci Report Request zaškrtněte položku Standard Report
- Potvrďte OK
- Totéž proveďte pro položku PocetCekPod3Min
- Zavřete dialog tlačítkem Close

Poznámka: v tomto stádiu výstavby našeho simulačního modelu však atributy PocetCekNad3Min a PocetCekPod3Min mají během i po ukončení simulačního běhu nedefinovanou hodnotu. Toto samozřejmě vyřešíme v dalších krocích.

9.3. Zobrazení standardního statistického přehledu

Nyní si řekneme, jakým způsobem si můžeme zobrazit standardní statistický přehled. Nejprve ale musíte spustit simulační běh. Po chvíli můžete simulační běh zastavit, ačkoli v našem případě bude pro získání vyhodnotitelných údajů nutno nechat proběhnout simulaci až do konce (pro urychlení můžete vypnout zobrazování animací, tj deaktivovat tlačítka



v horním panelu).

Po zastavení nebo dokončení simulačního běhu zadejte z menu příkaz Report / Display Standard Report. Dále doporučuji v sekci Display In zvolit položku Spread Sheet a případně nastavit cestu k tabulkovému procesoru MS Excel pomocí tlačítka Browse (pokud zde již cesta není správně přednastavena). Tlačítkem Display Report si poté můžete zobrazit statistické výsledky pro danou replikaci (např. Replication 1), nebo součtované či zprůměrované hodnoty při vícečetné replikaci.

9.4. Sledování vlastních statistických ukazatelů

Ačkoli nám SIMPROCESS umožní sledovat celou řadu statistických ukazatelů prostřednictvím standardního statistického přehledu, můžeme mít požadavky na vyhodnocení specifických ukazatelů. V takovém případě je opět nutno využít EXPRESSION SCRIPT a uživatelsky definované proměnné, tedy atributy.

Typickým příkladem v našem případě je sledování hodnot PocetCekNad3Min a PocetCekPod3Min. Nyní si ukážeme, jak toho docílit.

Před prvním použitím jakékoli proměnné (v našem případě atributů modelu či entit) je obvykle nutno provést její inicializaci. Prvotní inicializace atributů na začátku simulačního běhu je vhodné provést v EXPRESSION SCRIPT, který umístíte do události Start Trial() libovolného PROCESS či ACTIVITY, jež jsou součástí našeho modelu. Je však vhodné dodržet konvenci, že využijeme událost Start Trial() prvního objektu na nejvyšší úrovni v hierarchii modelu. V našem případě to je tedy PROCESS PrichodyZakazniku.

Postupujte tedy takto:

- Zobrazte si Properties procesu PrichodyZakazniku
- Stiskněte tlačítko Expressions ...
- Označte položku Start Trial () a stiskněte Edit ...
- Vložte kód scriptu uvedený v tabulce níže
- Potvrďte všechna otevřená dialogová okna

Tento postup zajistí inicializaci tří atributů (Model.PocetCekPod3Min, Model.PocetCekNad3Min, Model.CelkDobaCekani) na hodnotu nula.

Poznámka: kód níže uvedeného scriptu obsahuje také příkazy k aktualizaci hodnot dynamických textů. Jimi se budeme zabývat v dalších odstavcích.

Proces PrichodyZakazniku - StartTrial()

```
Model.PocetCekPod3Min:= 0;  
Model.PocetCekNad3Min:= 0  
Model.CelkDobaCekani:= 0.0;  
  
UpdateDynamicLabel  
(MasterEditor, "dlPocetCekNad3Min", 0, "Blue", Model.PocetCekNad3Min);  
  
UpdateDynamicLabel  
(MasterEditor, "dlPocetCekPod3Min", 0, "Blue", Model.PocetCekPod3Min);  
  
UpdateDynamicLabel  
(MasterEditor, "dlProcentoCekPod3Min", 0, "Blue", 0.0);  
  
UpdateDynamicLabel  
(MasterEditor, "dlPrumDobaCekani", 0, "Blue", 0.0);  
  
UpdateDynamicLabel  
(MasterEditor, "dlCelkPocetObslZak", 0, "Blue", 0);
```

Dále je třeba zajistit, abychom mohli vyhodnotit dobu čekání na obsluhu pro každého obsluhovaného zákazníka. K tomu nám slouží atributy ENTITY Zakaznik, konkrétně atribut ZacatekCekaniNaObsluhu a KonecCekaniNaObsluhu.

Zaznamenání počátku čekání na obsluhu u každého zákazníka provedeme např. takto:

- Vyvolejte dialog Properties pro PROCESS ObsluhaZakazniku
- Stiskněte tlačítko Expressions ...
- Označte položku s názvem události Accept Entity() a stiskněte tlačítko Edit ...
- Zadejte kód scriptu uvedený v tabulce níže
- Potvrďte všechna dialogová okna

Proces ObsluhaZakazniku - AcceptEntity()

```
Entity.ZacatekCekaniNaObsluhu:= SimTime;
```

Zaznamenání ukončení čekání na obsluhu (tj. čas počátku vlastní obsluhy) provedeme takto:

- Zobrazte si podprocesy procesu ObsluhaZakazniku
- Vyvolejte dialog Properties pro DELAY ACTIVITY ObsluhaNaBankomatuA
- Stiskněte tlačítko Expressions ...
- Označte položku s názvem události Get Resource() a stiskněte tlačítko Edit ...
- Zadejte kód scriptu uvedený v tabulce níže
- Potvrďte všechna dialogová okna
- Totéž proveďte pro DELAY ACTIVITY ObsluhaNaBankomatuB a ObsluhaNaBankomatuC

Aktivita ObsluhaNaBankomatuA/B/C - GetResource()

```
Entity.KonecCekaniNaObsluhu:= SimTime;
```

Poté, co zákazník ukončí obsluhu, je třeba aktualizovat atribut Model.CelkDobaCekani a dále počty zákazníků, kteří čekali méně a více než tři minuty. Postup je následující:

- Vyvolejte dialog Properties pro PROCESS ObsluhaZakazniku
- Stiskněte tlačítko Expressions ...
- Označte položku s názvem události Release Entity() a stiskněte tlačítko Edit ...
- Zadejte kód scriptu uvedený v tabulce níže
- Potvrďte všechna dialogová okna

Proces ObsluhaZakazniku - ReleaseEntity()

```
CekaniNaObsluhu: REAL;
ProcentoCekPod3Min: REAL;

CekaniNaObsluhu:= Entity.KonecCekaniNaObsluhu -
Entity.ZacatekCekaniNaObsluhu;

Model.CelkDobaCekani:= Model.CelkDobaCekani + CekaniNaObsluhu;

IF CekaniNaObsluhu < 180.0
    Model.PocetCekPod3Min:= Model.PocetCekPod3Min + 1;
ELSE Model.PocetCekNad3Min:= Model.PocetCekNad3Min + 1;
ENDIF;

ProcentoCekPod3Min:= 100.0 * FLOAT(Model.PocetCekPod3Min) /
    FLOAT(Model.PocetCekPod3Min + Model.PocetCekNad3Min);

UpdateDynamicLabel
(MasterEditor, "dlPocetCekNad3Min", 0, "Blue", Model.PocetCekNad3Min);

UpdateDynamicLabel
(MasterEditor, "dlPocetCekPod3Min", 0, "Blue", Model.PocetCekPod3Min);

UpdateDynamicLabel
(MasterEditor, "dlProcentoCekPod3Min", 0, "Blue", ProcentoCekPod3Min);

UpdateDynamicLabel
(MasterEditor, "dlPrumDobaCekani", 0, "Blue", Model.CelkDobaCekani /
    (FLOAT(Model.PocetCekNad3Min + Model.PocetCekPod3Min)*60.0));


UpdateDynamicLabel
(MasterEditor, "dlCelkPocetObslZak", 0, "Blue", Model.PocetCekNad3Min +
Model.PocetCekPod3Min);
```

10. Nápis a dynamický text (BACKGROUND TEXT / DYNAMIC LABEL)

Prostřednictvím objektu BACKGROUND TEXT nám SIMPROCESS nabízí možnost vkládat na plochu modelu statické nápisy a také text, který můžeme měnit v průběhu simulačního běhu. To se nám samozřejmě hodí v okamžiku, kdy chceme zobrazovat aktuální hodnotu nějaké proměnné (atributu).

10.1. Vložení statického textu

Zkuste si vložit na plochu modelu statický text takto:

- Přejděte na nejvyšší úroveň v hierarchii modelu
- Klikněte na ikonu BACKGROUND TEXT 
- Klikněte na ploše modelu - uprostřed v její horní části
- V nově otevřeném dialogu vyplňte pole Static Label textem "Demonstrační simulační projekt"
- V sekci Font Attributes vyberte Size 36 a Color Blue
- Zaškrtněte položku Bold
- Potvrďte OK
- Stejným způsobem vložte text "EM-Banka", velikost 48, barva Red, tučně
- Operací drag-and-drop zarovnejte vložený text dle potřeby

10.2. Použití dynamického textu

Řekněme, že budeme chtít během simulačního běhu zobrazovat aktuální hodnoty určitých statistických atributů, např. počet zákazníků, kteří museli čekat méně než tři minuty. Právě to nám umožní objekty dynamického textu. Postupujte prosím takto:

- V menu Layout zaškrtněte položku Grid Lines a Snap To Grid
- Postupem popsaným v kapitole 10.1 vložte na plochu modelu všechny statické texty, které vidíte na obrázku níže, kromě nul zobrazených modrou barvou, přičemž použijte příslušnou barvu a font Arial velikosti 12, tučně

Demonstrační simulační projekt EM-Banka



Statistika zákazníků

Počet čekajících pod 3 minuty:	0	Průměrná doba čekání v minutách:	0
Počet čekajících nad 3 minuty:	0	Celkový počet obslužených zákazníků:	0
Procento čekajících pod 3 minuty:	0		

Následně vpravo vedle textu "Počet čekajících pod 3 minuty" vložte objekt statický text, avšak v okamžiku, kdy se vám zobrazí dialogový rámeček pro vyplnění jeho vlastností, proveďte navíc tyto kroky:

- Položku Name vyplňte hodnotou 0
- V sekci Font Attributes nastavte navíc Color na Blue
- V sekci Dynamic Label Properties vyplňte Name textem dlPocetCekPod3Min a roletku Mode nastavte na Integer, hodnotu ID zadejte 0

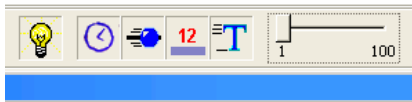
Stejným způsobem vložte na příslušné pozice další dynamické texty s těmito parametry:

- Name: dlPocetCekNad3Min, Mode: Integer
- Name: dlProcentoCekPod3Min, Mode: Real
- Name: dlPrumDobaCekani, Mode: Real
- Name: dlCelkPocetObslZak, Mode: Integer

Vlastní aktualizaci zobrazované hodnoty během simulačního běhu provedete v příslušném EXPRESSION SCRIPT voláním funkce UpdateDynamicLabel(...), což však již máme v našich scriptech ošetřeno.

11. Závěrečné testování

V tomto okamžiku by již náš simulační model měl obsahovat vše požadované. Můžete tedy provést závěrečné testování funkčnosti. Pokud si v horním panelu zapnete tlačítka



, měli byste po spuštění simulačního běhu kromě animace pohybujících se entit a zobrazování počtu entit nacházejících se v daném procesu mít možnost sledovat aktuální hodnoty dynamických textů.

Vzhledem k tomu, že chování našeho simulačního modelu má stochastický charakter, pro získání statistických výsledků majících vypovídající hodnotu budete muset spustit simulační běh ve více replikacích a poté vyhodnocovat průměry a odchylky získaných výsledků. Spuštění N replikací simulačního běhu nastavíte pomocí příkazu menu Simulate / Run Settings ..., přičemž v otevřeném dialogovém okně nastavíte položku Number of Replications na požadovanou hodnotu N.