

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ  
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Modelování a simulace – Projekt  
SHO v logistice

# **1 Úvod**

## **1.1 Téma**

Projekt rieši modelovanie [1] (s. 8) a simuláciu [1] (s. 8) tzv. systému hromadnej obsluhy [1] (s. 136) (ďalej len "SHO") v oblasti logistiky. Za riešený SHO bol vybraný logistický systém prijímania a vybavovania objednávok v reštaurácii Pizza23 v meste Nové Zámky (SR).

Systém rieši prijímanie objednávok, prípravu jedla, balenie objednávok a následnú dopravu zákazníčkovi.

## **1.2 Zdroje informácií**

Hlavnými zdrojmi pre obsah SHO boli informácie od zamestnancov danej pobočky podniku Pizza23. Zdrojmi informácií pre implementačné riešenie boli výukové materiály predmetu Modelování a simulace (VUT FIT v Brně) a dokumentácia knižnice SIMLIB.

## **1.3 Validita informácií**

Informácie boli získané od zamestnancov reštaurácie, avšak vzhľadom na dĺžku praxe opýtaných zamestnancov a stálosť časových údajov systému sa tieto informácie dajú považovať za dôveryhodné a použiteľné pre model systému.

## **1.4 Validita modelu**

Údaje získané pomocou experimentov vykonaných nad vytvoreným modelom vo veľkej miere odpovedajú aktuálnemu stavu skúmaného systému, to znamená, že vytvorený model môžeme považovať za validný - spĺňa validitu modelu [1] (s. 37).

## 2 Koncepcia modelu

### 2.1 Parametre

Medzi sledované parametre systému, ktoré budeme sledovať v experimentoch patria : počet kuchárov, počet rozvozárov, trvanie prípravy jedla a trvanie dopravy online objednávky.

### 2.2 Petriho sieť modelu

Petriho sieť modelu popisuje proces objednávky. pričom reštaurácia má systém pre osobné a systém pre online objednávky. Obidva druhy objednávok sú generované jedným generátorom a rozdeľujú sa na základe percentuálneho podielu - osobné objednávky tvoria cca. 60% a online objednávky 40%;

Procesy sa generujú v určitých intervaloch. Jeden vygenerovaný proces označuje 1 objednávku, avšak 1 objednávka sa nerovná 1 jedlu, ale celému zoznamu objednaného jedla z danej objednávky. (Kuchár, ktorý si danú objednávku berie, ju väčšinou pripravuje naraz.)

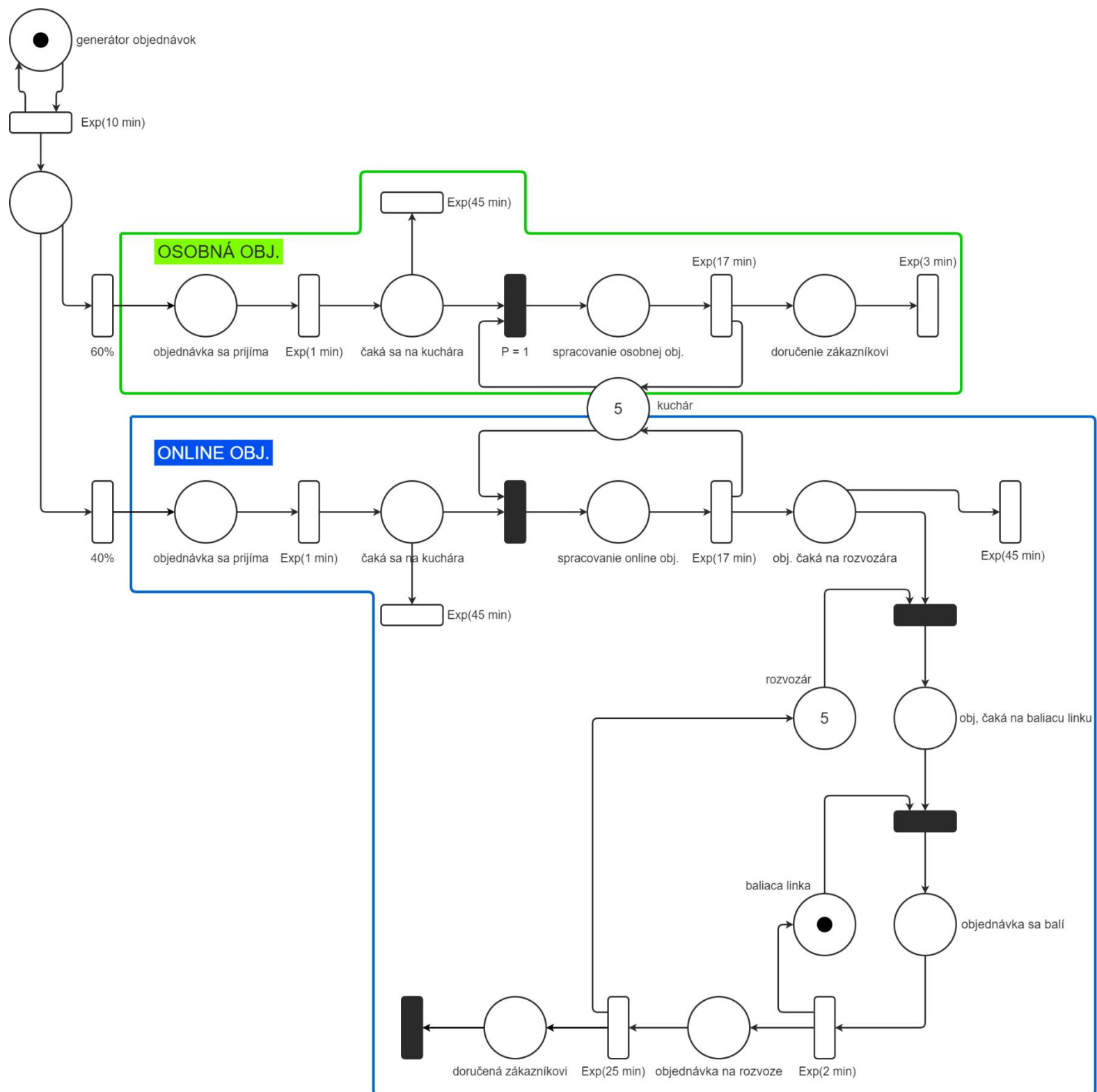
Časť *osobná obj.* popisuje proces objednávky objednanej priamo v reštaurácii. Táto objednávka zahŕňa prijatie čašníkom, prípravu jedla a následné servírovanie zákazníčkovi.

Časť *online obj.* popisuje objednávku prijatú cez webovú stránku reštaurácie. Od osobnej sa odlišuje nutnosťou zabalenia jedla, kvôli jeho udržaniu počas prepravy a dopravou objednávky na určené miesto.

Samotní čašníci vybavujúci osobné objednávky nie sú v modeli systému, pretože nie sú pre účely experimentov akokoľvek potrební.

Obidva druhy objednávok zdieľajú spoločnú časť modelu, ktorou sú samotní kuchári, keďže jedlo je pripravované na jednom mieste (v kuchyni).

Zároveň obidva procesy sú modelované s prihliadnutím na netrpezlivosť zákazníčkov a po určitej dobe po vybavení objednávky opúšťajú systém, ak nie sú prevzaté jedným z kuchárov, alebo v prípade ak nie sú prevzaté rozvozárom. Online objednávky majú o niečo vyššiu trpezlivosť, než osobné, keďže v ich prípade môže vzniknúť čakanie v dvoch miestach systému.



## 3 Simulačný model

### 3.1 Prevod na simulačný model

Prevod na simulačný model [1] (s. 44) bol vykonaný pomocou najnovšej verzie SIMLIB-u [2] v jazyku C++. Simulačný model je obsiahnutý v súbore `ims.cpp`.

Program sa dá spustiť :

- v základnom režime (so zistenými hodnotami) ako : `make run`
- s nastavením počtu kuchárov : napr. `./ims -k 8`
- s nastavením počtu rozvozárov : napr. `./ims -r 9`
- s nastavením doby prípravy jedla : napr. `./ims -j 15`
- s nastavením doby dopravy jedla : napr. `./ims -d 20`
- s nastavením časového rozostupu objednávok : napr. `./ims -f 10`
- s nastavením trvania simulácie (v min.) : napr. `./ims -t 1440`

Parametre sa dajú ľubovoľne kombinovať.

### 3.2 Popis implementovaného modelu

Model obsahuje dva procesy: `OsobnaObjednavka` a `OnlineObjednavka`. Kuchári a rozvozári (čiže pracovníci) sú navrhnutí cez tzv. `Store()`, pričom kapacita každého `Store()` predstavuje počet pracovníkov a je možné ju meniť cez parametre programu. Obslužná baliaca linka je implementovaná ako `Facility BaliacaLinka`.

Generovanie prebieha v jednom generátori a pravdepodobnostné rozdelenie je implementované pomocou funkcie `Random()`, ktorá generuje pseudonáhodné číslo v intervale 0-1 a na základe hodnoty sa vygeneruje proces osobnej alebo proces online objednávky.

Metóda `Behavior()` v oboch procesoch implementuje ich chovanie, teda činnosti, ktoré vykonávajú. Využívajú sa funkcie `Enter()` a `Leave()`, ktoré popisujú zaberanie a uvoľňovanie kuchárov alebo rozvozárov. Pre obslužnú linku sa za tým istým účelom využívajú metódy `Seize()` a `Release()`.

Trieda `Timeout()` implementuje časovač čakania procesov v troch miestach systému - pri čakaní na kuchára pre obidva druhy procesov a pri čakaní na rozvozára pre proces online obj. Pri dosiahnutí vopred definovaných časov sa proces odstráni, t.j. opustí systém.

Všetky časové údaje sú obsiahnuté v definovaných premenných a sú implementované s exponenciálnym [1] (s. 91) rozdelením. Tieto časové údaje sú využívané v metóde `Wait()`, ktorá je použitá na označenie doby vykonávania určitej činnosti, alebo pri časovom rozstupe generovaných procesov v modeli.

## 4 Simulácie

### 4.1 Popis experimentov

Pred návrhom experimentov sme simulovali chod systému so skutočnými získanými údajmi a sledovali sme jednotlivé prvky systému a ich chovanie. Medzi sledované údaje patrí hlavne vyťaženie pracovníkov, t.j. ako stíhajú spracovávať objednávky v jednotlivých miestach systému, vyťaženie baliacej linky a počet nespokojných zákazníkov.

### 4.2 Motivácia experimentov

V realite je skúmaný systém funkčný, ale vzhľadom na momentálnu ekonomickú situáciu sa reštaurácia snaží ušetriť finančné prostriedky vo všetkých bodoch systému. Vlastnosti ako cena jedál, alebo výplata pracovníkov v tomto modeli skúmať nebudeme, ale je možné pracovať so zmenou počtu zamestnancov.

Vzhľadom na to, že znižovanie počtu zamestnancov na úplné prijateľné minimum je bežná praktika vo viacerých firmách, tak môžeme zistiť, či je v skúmanom systéme možné vykonávať takéto zmeny (a v akej miere).

Pre účely experimentov využijeme počet zrušených objednávok, ktorý značí nasledujúce:

- osobné objednávky čakajúce na kuchára aspoň 45 min.
- online objednávky čakajúce na kuchára aspoň 45 min.
- hotové online objednávky čakajúce na rozvozára aspoň 45 min.

Pri online objednávkach majú zákazníci k dispozícii sledovanie objednávky v reálnom čase a zvyčajne rušia objednávku po 45 min. od prijatia, alebo po 45 min. od hotovej objednávky čakajúcej na odvoz.

## 4.3 Postupy experimentov

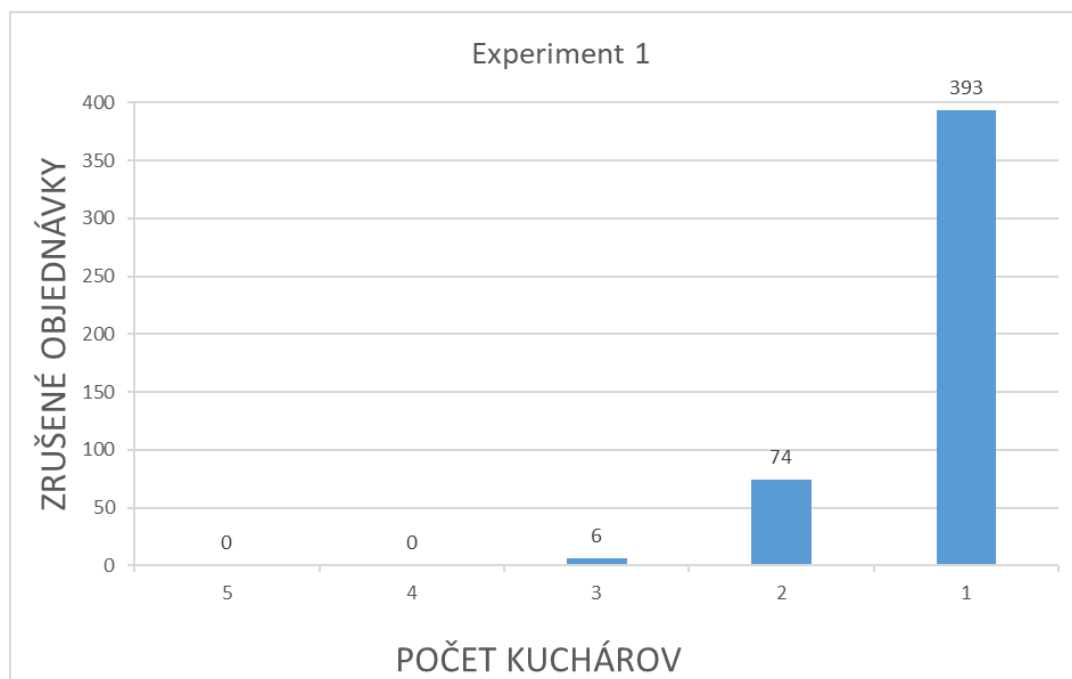
### 4.3.1 Experiment 1

Príkaz: `./ims -k K`, pričom  $K \in \langle 0, 5 \rangle$ ,  $K \in \mathbb{Z}$

V prvom experimente upravujeme hodnotu počtu kuchárov a to postupným znižovaním z reálnej hodnoty. Sledujeme hlavne množstvo zrušených objednávok - čiže objednávky, ktoré čakali na kuchára viac, než 45 minút (bez ohľadu na typ objednávky)

Zisťujeme, že.

- Systém funguje v poriadku pri aktuálnom množstve kuchárov.
- Pri znížení počtu kuchárov na 3 vzniká v systéme vysoké čakanie na objednávky a ich rušenie.



Obr. 1: Experiment 1

Vzhľadom na získané údaje sme zistili, že znižovaním počtu kuchárov sa predlžuje čakanie na objednávku a pri určitom počte kuchárov sa pri niektorých objednávkach predlžuje na neprijateľné hodnoty. To vedie k postupne rastúcemu množstvu zrušených objednávok.

Pokiaľ je reštaurácia ochotná akceptovať možných nespokojných zákazníkov, tak je možné znížiť počet na menšiu hodnotu, pričom hodnoty zrušených objednávok sú uvedené v grafe č. 1.

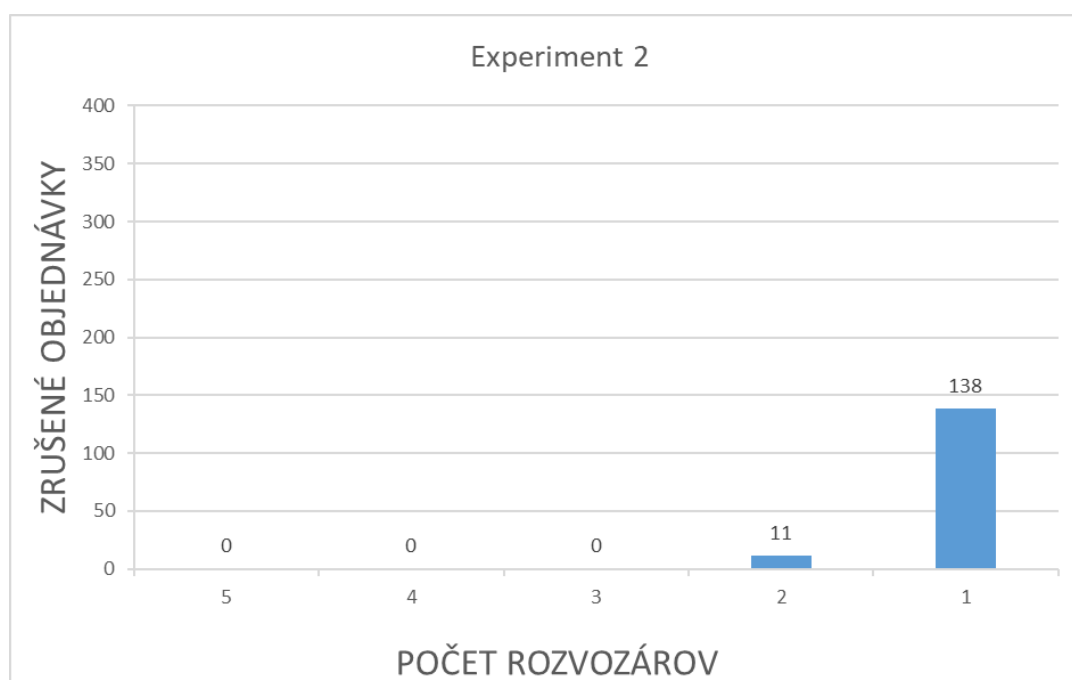
### 4.3.2 Experiment 2

Príkaz: `./ims -r R`, pričom  $R \in \langle 0, 5 \rangle, R \in \mathbb{Z}$

V druhom experimente upravujeme hodnotu počtu rozvozárov a to postupným znižovaním z reálnej hodnoty. Sledujeme znovu počet zrušených objednávok.

Zisťujeme, že.

- Systém funguje pri aktuálnom množstve rozvozárov.
- Pri znížení počtu rozvozárov na 2 sa začínajú vyskytovať zrušené objednávky.



Obr. 2: Experiment 2

Vzhľadom na získané údaje sme zistili, že aj znižovaním počtu rozvozárov sa predlžuje čakanie na objednávku a rastie množstvo zrušených objednávok. Avšak pri rozvozároch to platí až na počte 2 rozvozárov,



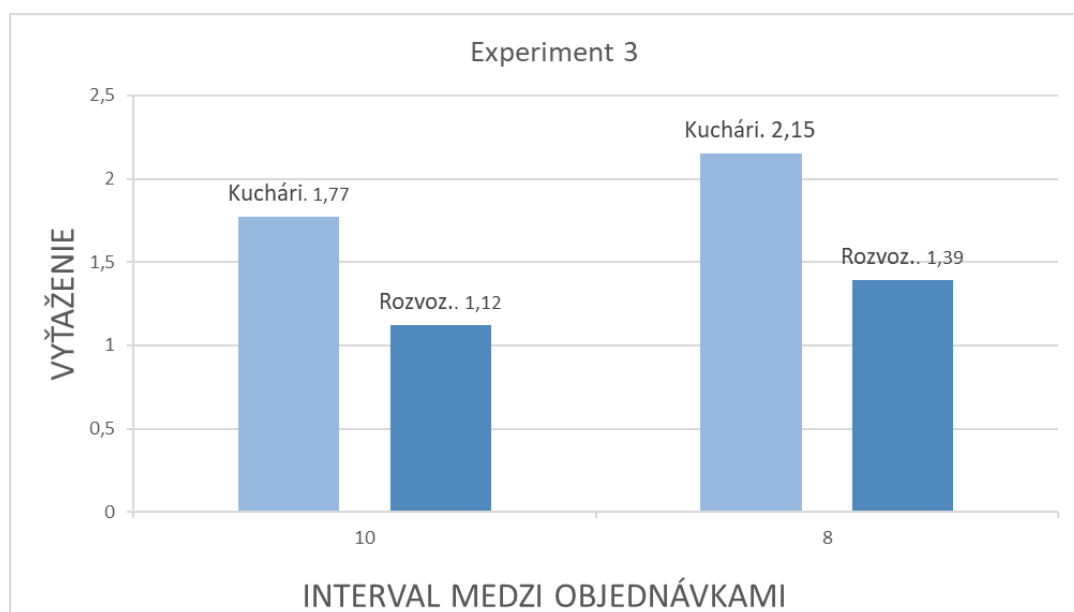
### 4.3.3 Experiment 3

Príkaz `./ims -f F`, pričom  $F = 8$

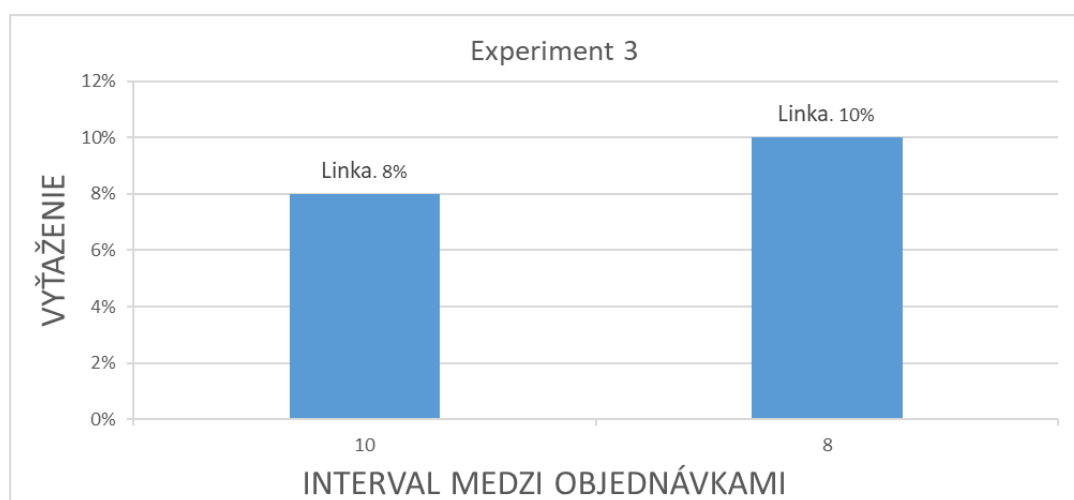
V treťom experimente testujeme chovanie systému počas kritických období, napr. začiatok zimného obdobia, kedy je záťaž systému o niečo vyššia.

Na základe reálneho fungovania systému vieme, že zvláda zvýšený nápor zákazníkov, ale hypotézu si overíme pomocou experimentu nad modelom.

Zvýšenie záťaže sa dá jednoducho napodobniť zvýšením frekvencie generovaných objednávok. Pre experiment znížime časový rozstup generovaných objednávok na 8 min.



Obr. 3: Experiment 3



Obr. 4: Experiment 3 - 2.časť

Sledovaným údajom ukazujúcim vyťaženie bolo priemerné vyťaženie na troch miestach systému. Na obrázku 3 a 4 vidieť, ako sa vyťaženie zvýšilo pri zvýšení frekvencie príchodu objednávok (čo aj bolo predpokladané). Avšak zistili sme, že aj počas zvýšenej záťaže je systém schopný spracovávať objednávky bez vzniku neprijateľných čakaní, alebo bez neúnosnej záťaže zamestnancov, pričom problémy tohto typu by mohli viesť k nespokojnosti ako zamestnancov, tak aj zákazníkov.

## 5 Záver

Cieľom tohto projektu bolo previesť reálny systém do abstraktného modelu za účelom konkrétnych experimentov.

Prvé dva experimenty sa venovali skúmaniu systému a určitých zmien systému za účelom úpravy podmienok reálneho systému v budúcnosti (ale vyvodenie konkrétnych záverov a uskutočnenie zmien je mimo našich kompetencií).

Tretí experiment sa venoval bežnému problému zvýšenej záťaže systému v určitom období a jeho cieľom bolo len potvrdiť existujúce slovné hypotézy na základe experimentov so správne vytvoreným abstraktným modelom.

## Literatúra

[1] Peringer P., Hrubý M., *Modelování a simulace*, 2021

<http://www.fit.vutbr.cz/study/courses/IMS/public/prednasky/IMS.pdf>

[2] Peringer P., *SIMulation LIBrary for C++*, 1991

<http://www.fit.vutbr.cz/~peringer/SIMLIB/.cs>