# **Návrh**

# **Obsah**

[**1. Úvod**](#_x9zwrjaq49ee) **2**

[1.1 Účel dokumentu](#_jxgvf6e3fvcy) 2

[1.2 Zameranie a rozsah](#_jxgvf6e3fvcy) 2

[1.3 Slovník pojmov](#_4c88nejqlude) 2

[**2. Rozdelenie:**](#_f0ky5r67n5ep) **3**

[2.1. Komponenty](#_for0q9gjnu0z) 3

[2.2. Súbežné procesy](#_to81jhknn09z) 6

[**3. Používateľské rozhranie**](#_l3j7t9sj9xae) **9**

[3.1. Prvá fáza konfigurácie](#_mkf482pnt4hh) 9

[3.2. Druhá fáza konfigurácie](#_ebshwql2y6bc) 10

[3.3. Režim behu](#_j51dalp3sypf) 11

[**4. Závislosti**](#_oqz1fjyp22at) **12**

[4.1. Externé rozhrania](#_m7gbh8fbv00f) 12

[4.2. Perzistentné súbory](#_tmcy8ipgnawm) 13

[**5. Implementácia:**](#_r3vgpegxz5hx) **14**

[5.1. Triedny diagram](#_71jm13xgbusd) 14

[5.2. Popis tried](#_c69aszv356c1) 14

[5.3. Algoritmus spracovania obrazu](#_2zu0vybgb67j) 16

[5.4. Cieľové prostredie](#_pisf5tms2qh6) 16

# **1. Úvod**

## 1.1 Účel dokumentu

Tento dokument predstavuje podrobný popis návrhu systému fotopasce. Pomocou diagramov a detailných popisov je tu vysvetlené, akým spôsobom bude systém vyvinutý a ako bude následne fungovať tak, aby splnil všetky požiadavky uvedené v Katalógu požiadaviek.

## 1.2 Zameranie a rozsah

Predpokladáme, že čitateľ tohto dokumentu má už prečítaný katalóg požiadaviek. Tam je totiž podrobne vysvetlené, *čo* má systém robiť a tu už sa zaoberáme len otázkou *ako*. Veľkú časť tohto dokumentu tvoria UML diagramy, v ktorých je podrobne znázornené, na aké moduly a triedy bude systém rozdelený a ako spolu jednotlivé časti súvisia. V časti o súbežných procesoch uvádzame aj sekvenčný diagram, ktorý znázorňuje, akým spôsobom prebieha snáď najkritickejšia časť celého systému, a to kontrola pohybu vo vyznačených oblastiach. Samostatnú podkapitolu sme sa rozhodli venovať aj návrhu používateľského rozhrania.

## 1.3 Slovník pojmov

Niekoľko pojmov, ktoré sa vyskytujú v tomto dokumente:

* **frame/snímka:** oba tieto výrazy vyjadrujú jeden konkrétny záber z výstupu webkamery a používame ich ako synonymum
* **komponent:** vyjadruje ucelenú časť systému, ktorú môže tvoriť viacero tried. Výraz modul má v podstate rovnaký význam, ale kladie väčší dôraz na fakt, že má byť komponent s takýmto označením modulárny/vymeniteľný
* **vlákno:** nezávisle bežiaci proces/činnosť v rámci systému, ktorý plní nejakú úlohu (teda nebeží zbytočne)
* **log súbor:** súbor s textovým záznamom o nejakej udalosti, obsahuje dátum, čas a popis
* **ukladač:** komponent starajúci sa o ukladanie snímkov a videí na disk
* **logovač:** komponent zapisujúci jednotlivé udalosti do log súboru

1.4 Prehľad nasledujúcich kapitol

V 3. kapitole uvádzame komponentný diagram aj s popisom, ako bude systém rozdelený na jednotlivé komponenty. Ďalej je tu popis súbežných procesov a sekvenčný diagram, ktorý ich znázorňuje v kontexte behu systému. V 4. kapitole možno nájsť náčrty grafického uživateľského rozhrania častí systému. Systém bude závislý na niekoľkých externých knižniciach, s ktorými komunikuje cez rozhrania, teda prostredníctvom ich metód. Ich stručný zoznam uvádzame v kapitole 5., spoločne s popisom perzistentných súborov systému. Nakoniec v 6. kapitole je podrobný návrh implementácie systému, spoločne s triednym diagramom a popisom cieľového prostredia prevádzky.

# **2. Rozdelenie:**

## 2.1. Komponenty

Systém sme rozdelili na 4 časti, každá časť v sebe obsahuje ešte ďalšie, menšie komponenty.

2.1.1. GUI

* Prvý modul je GUI, ktorý v našom systéme má menšie komponenty, ktoré sú rozdelenie na fázy konfigurácie, v prvej fáze sa nám zobrazí okno “Konfigurácia”, kde sa pomocou tlačidiel a posuvných bežcov, bude dať nastaviť konfigurácia. Pre každý modul sú priradené tlačidlá na nastavenie modulu.V druhej fáze sa nám zobrazí okno “fotopasca”, v ktorom už uvidíme reálny záber z kamery a pomocou myši si vieme vybrať oblasť záujmu. Na spodnej lište sa nachádzajú súradnice vyznačenej oblasti, bežec na nastavenie citlivosti, tlačidlo na odstránenie oblasti.
  + **Modul obsahuje:**
    - GUI editor - Nastavenie kamery
    - GUI editor - Nastavenie snímania
    - Vizualizácia snímania
    - Alarmovač

2.1.2. Konfigurácia

* V tomto module sa budú nastavovať všetky dostupné a možné nastavenia pre kameru, ktoré slúžia na nastavenie systému a vizualizáciu snímania. Tu si bude môcť užívateľ nastaviť viac fáz konfigurácie, v prvej fáze si vieme nastaviť kameru (rozlíšenie, snímkovanie), ďalej režim (časozber, interval časozberu, nahrávanie pri poplachu), nastavenie alarmu (alarm pri poplachu, oneskorenie alarmu, doznenie alarmu) a nakoniec ostatné nastavenia (auto štart, heslo, oneskorenie štartu). Druhá fáza obsahuje nastavenie oblasti záujmu, tu si budeme môcť nastaviť ROI, ktoré režim behu bude sledovať.
  + **Modul obsahuje:**
    - Nastavenie kamery
    - Nastavenie oblasti záujmu
    - Nastavenie snímania

2.1.3. Režim behu

* Zaznamenáva a kontroluje snímky z webkamery. Frame, ktorý bude uchovávaný v troj-rozmernej matici, ďalej bude spracovávať obraz, spracovávať pohyb a ukladať ich v predpísanom formáte do úložiska, zaznamenávať časozberné video, prideľovať obrázky k log súborom a rozpoznávať pohyb. Kontrola bude kontrolovať pohyb, v prípade pohybu spustí alarm.
  + **Modul obsahuje:**
    - Režim behu
    - Kamera loop
    - Riadenie snímania
    - Kontrola

2.1.4. Úložisko

* Tento modul a jeho pod-komponenty sa budú starať o uchovávanie a spracovanie dát s ktorými sme pracovali a vďaka tomuto modulu k nim môžeme dodatočne pristupovať. Taktiež v sebe uchová nastavenia, ku ktorým môžeme pristupovať a dodatočne ich meniť/konfigurovať, ak sa jedna o uložené videa/obrázky môžme k ním pristupovať tiež.
  + **Modul obsahuje:**
    - Prácu s konfiguráciou
    - Ukladanie framov a videa
    - Logovač

## 

*diagram 1: komponentný diagram*

## 2.2. Súbežné procesy

Tu uvádzame popis kritických súčastí systému. V skutočnosti budú neustále v systéme bežať iba procesy Riadenie snímania a GUI (podrobnosti nižšie.) Hlavný je proces *Riadenie snímania*, ktorý ovláda ostatné časti a rozdáva im príkazy. Logicky sa ale aj ďalšie uvádzané procesy dajú chápať ako samostatné vlákna, pretože sú neustále k dispozicii a plnia úlohy počas celého behu systému. Najskôr uvádzame ich stručný popis a na druhej strane možno vidieť sekvenčný diagram, ktorý znázorňuje činnosť systému.

2.2.1. Riadenie snímania

V podstate hlavný proces celého systému, ktorý riadi ostatné komponenty v režime behu. Pamätá si zoznam vyznačených oblastí záujmu aj ich súradnicami, ktoré boli navolené v režime konfigurácie. V pravidelných intervaloch oslovuje proces *Kontrola,* ktorý preskúma, či v danej oblasti nedošlo k pohybu, ktorý prekračuje povolenú citlivosť.

2.2.2. Kontrola

Proces je v pripravený na to, aby skontroloval, či sa v danej oblasti vyskytuje pohyb. Uchováva si predchádzajúci snímok a komunikuje priamo s výstupom z webkamery, ktorá mu posiela aktuálne zábery.

2.2.3. Kamera

V nakonfigurovanej frekvencii spracováva výstup z webkamery ako samostatné snímky a posiela ich v prípade potreby ostatným procesom.

2.2.4. Ukladač a Logovač

Oba tieto procesy sa starajú o ukladanie perzistentných súborov. Ukladač má svoju vyrovnávaciu pamäť, pretože občas prichádzajú požiadavky na uloženie snímkov a videí rýchlejšie, než je ich možné ukladať na disk.

2.2.5. Časozber

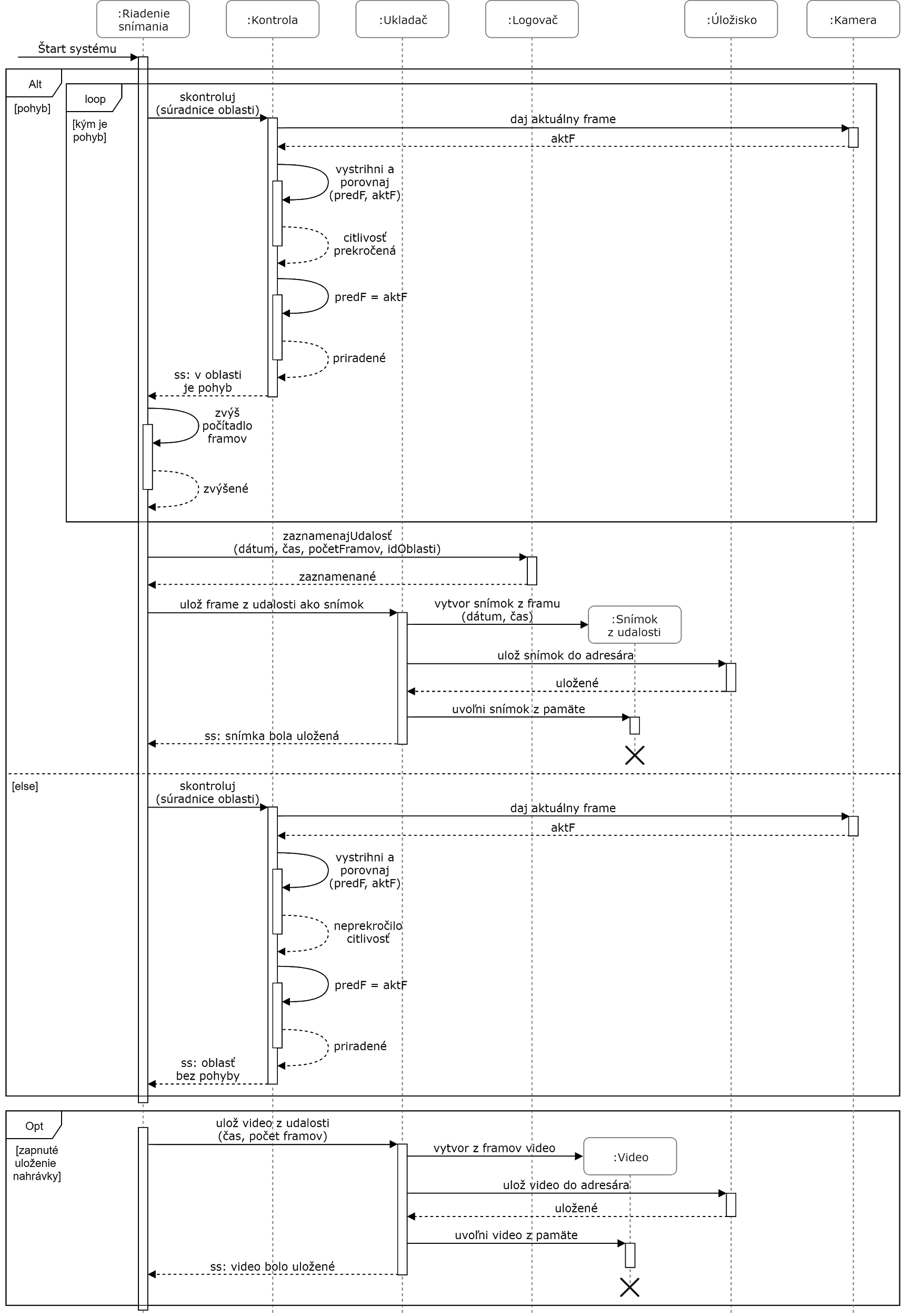
Ak bola v konfigurácii zapnutá funkcia časozberu, tak počas behu systému je aktívne vlákno príslušného komponentu, ktorý v pravidelných intervaloch ukladá snímky.

2.2.6. GUI

Okrem Riadenia systému má svoje vlákno (respektíve *while loop*) aj grafické uživateľské rozhranie, ktoré reaguje na podnety od užívateľa. Samotné spracovanie ale už má na starosti *Riadenie snímania.*

2.2.6. Sekvenčný diagram

Sekvenčný diagram na druhej strane zobrazuje beh systému, v ktorom sa pravidelne môžu vyskytovať dva alternatívne scenáre. *Riadenie snímania* dotazuje *Kontrolu* aby skontrolovala, či v oblasti s danými súradnicami nie je pohyb prekračujúci nastavenú citlivosť. Pokiaľ je v tejto oblasti pohyb, *Riadenie snímania* si zvyšuje vnútorné počítadlo, čím počíta počet snímkov, pokiaľ trvala táto udalosť. Následne *Logovač* túto udalosť zaznamená a *Ukladač* uloží príslušnú snímku. Ďalej je tu *optional* časť, ktorá prebieha, iba ak bola aktivovaná funkcia záznamu videa.

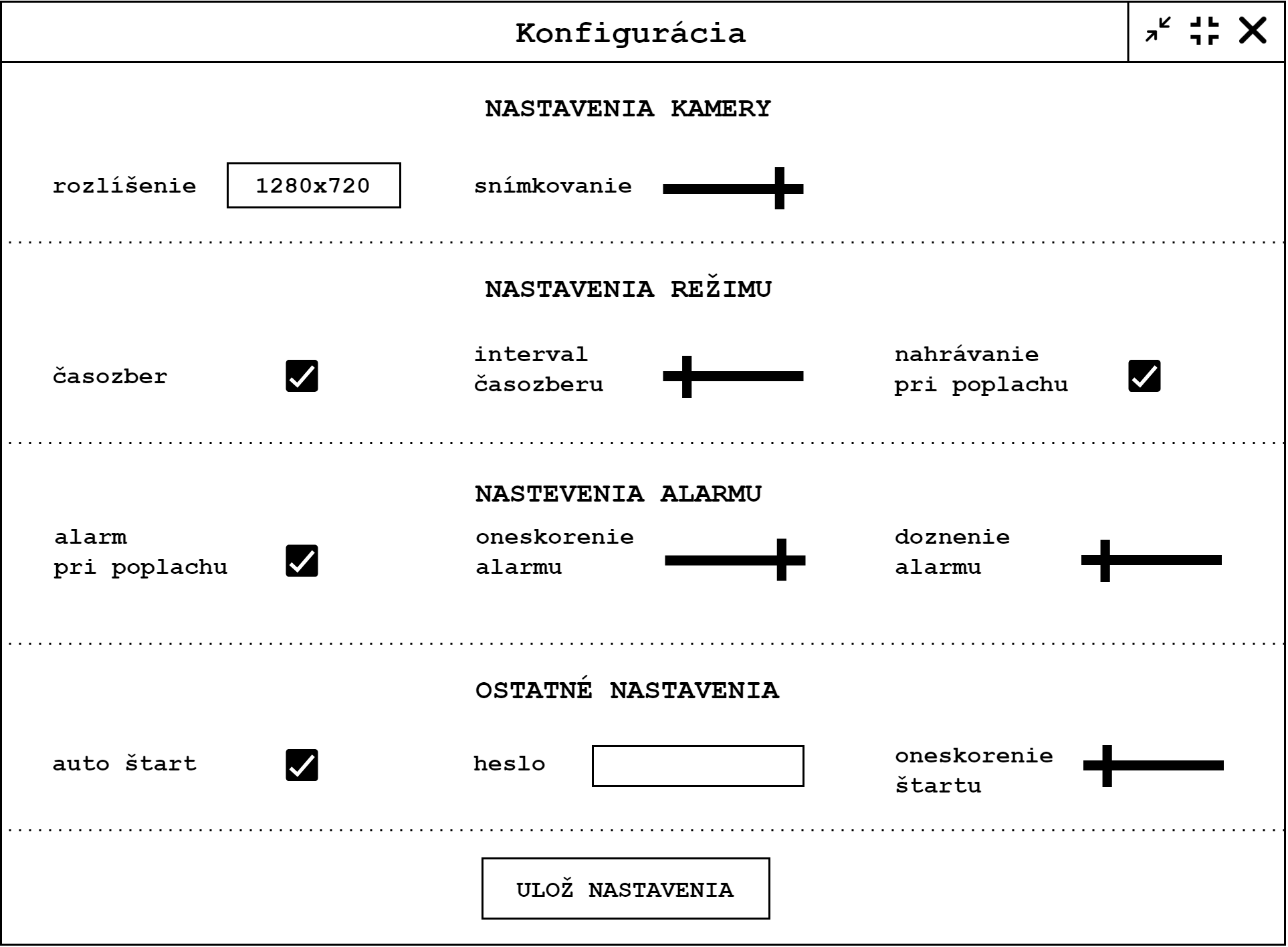


*diagram 2: sekvenčný diagram, zobrazujúci činnosť systému*

# **3. Používateľské rozhranie**

V tejto časti uvádzame náčrt uživateľského rozhrania, spoločne s popisom jednotlivých častí. Bližšie informácie k funkciám jednotlivých nastavení možno nájsť aj v katalógu požiadaviek, z ktorého návrh vychádza.

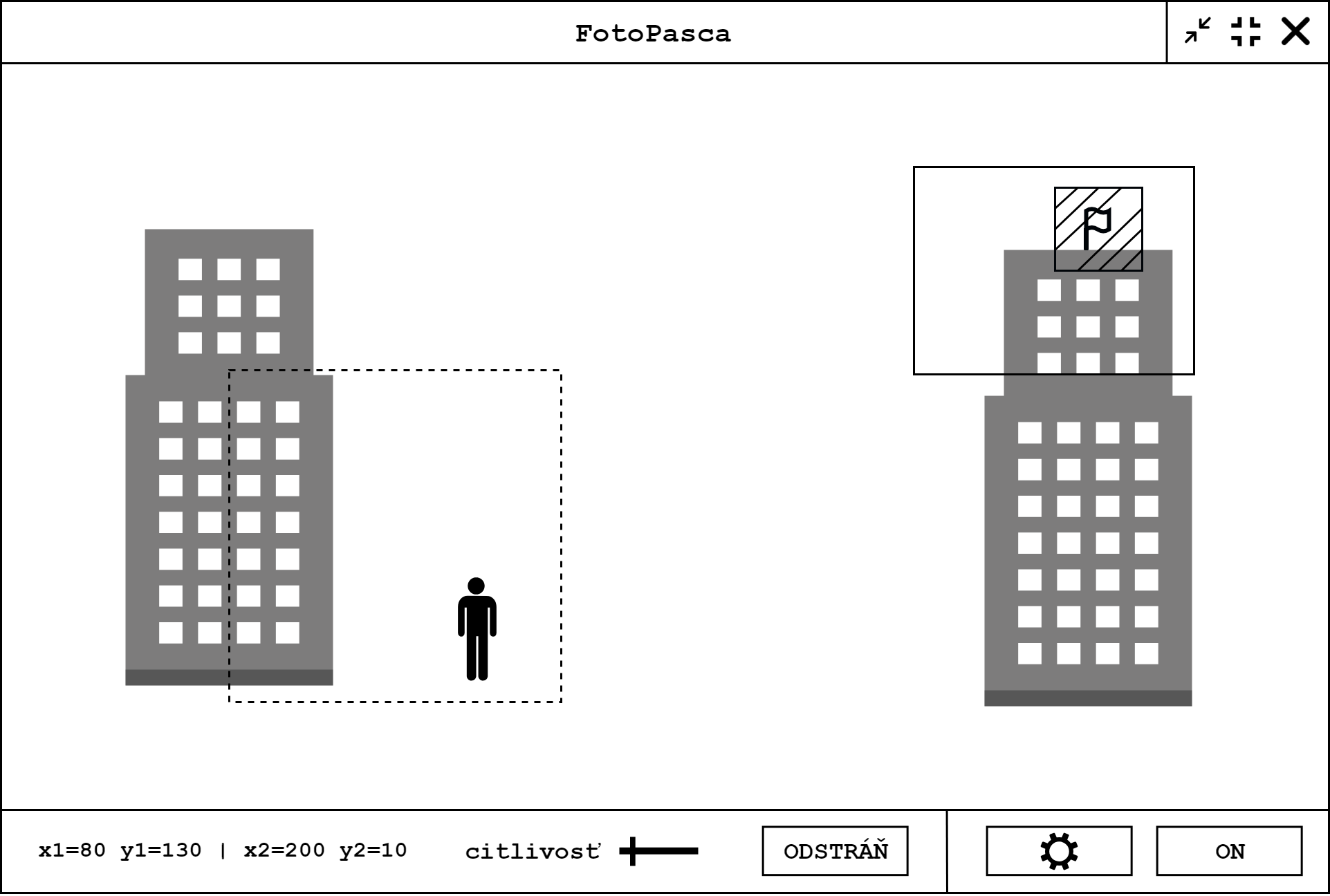
## 3.1. Prvá fáza konfigurácie



*Obr.1: náčrt prvej fázy konfigurácie*

Základné nastavenia, rozdelené do jednotlivých kategórií. Konkrétne intervaly hodnôt posuvných bežcov budú upresnené pri implementácii. Pri snímkovaní pôjde o *počet snímkov za sekundu* a pri všetkých ostatných bežcoch o hodnotu v milisekundách. Vstupné pole pre heslo môže klásť obmedzenie na dĺžku hesla. Po kliknutí na tlačidlo uloženia sa všetky nastavenia zaznamenajú v príslušnom konfiguračnom súbore.

## 3.2. Druhá fáza konfigurácie



*Obr.2: náčrt druhej fázy konfigurácie*

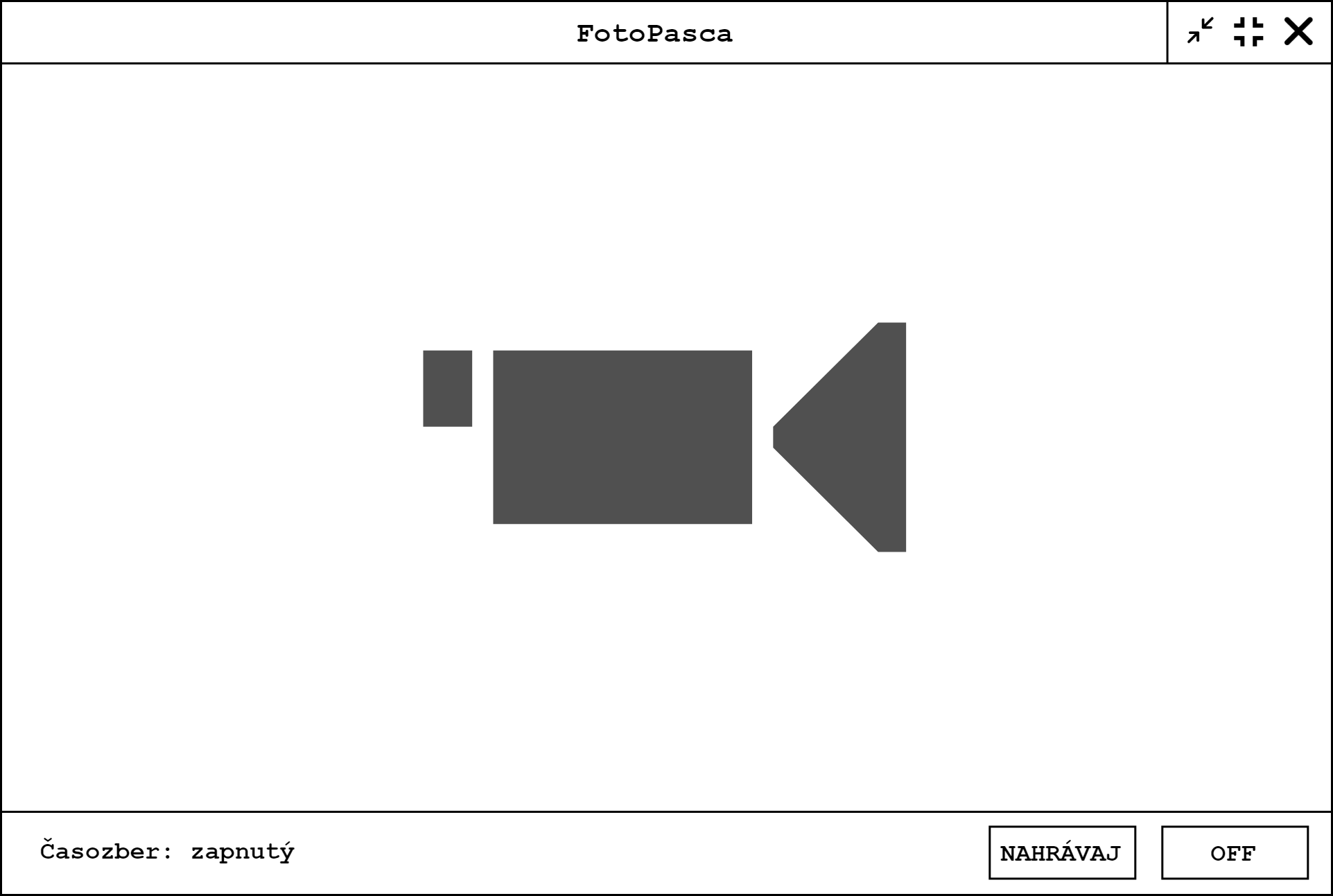
Po uložení nastavenia v prvej fázy sa zobrazí rozhranie druhej fázy. Z technických dôvodov sa na to pri implementácii buď vytvorí nové okno (pričom okno prvej fázy sa zavrie), alebo sa rozhranie načíta v rovnakom systémovom okne. Tu bude možné vyznačiť jednotlivé oblasti záujmu (ROI), ktoré budú v režime behu systémom sledované. Ich počet môže byť z technických dôvodov obmedzený. Na pozadí bude živý výstup z kamery.

Ignorované oblasti v rámci oblastí záujmu sa budú dať vytvoriť podľa špecifikácie z katalógu požiadaviek (pozri **B - RK11)**. Oblasť s prerušovaným okrajom na nákrese znázorňuje aktuálne vybranú oblasť (naposledy vytvorená, alebo zvolená kurzorom myši). Vľavo dole môžeme vidieť súradnice ľavého horného (*x1*, *y2*) a pravého dolného (*x2, y2*) okraju tejto oblasti. Ďalej tu môžeme nastaviť citlivosť na pohyb, ktorej hodnoty budú bližšie vysvetlené v manuáli s ukážkami. Kliknutím na tlačidlo môže používateľ túto oblasť odstrániť.

Nové oblasti záujmu sa dajú vytvoriť kliknutím a ťahaním myši. Tak, ako bolo vysvetlené aj v katalógu požiadaviek, oblasti sa budú ukladať na seba a teda ak sa nejaké dve oblasti prekrývajú, po kliknutí je vyznačená tá, ktorá bola pridaná neskôr.

Tlačidlo s ozubeným kolieskom v pravom dolnom rohu slúži na návrat do prvej fázy konfigurácie a tlačidlo ON na zapnutie systému a prechod do režimu behu.

## 3.3. Režim behu



*Obr.3: náčrt režimu behu*

Po prechode do režimu behu sa zobrazí rozhranie, ktorého hlavnú časť okupuje živý náhľad z kamery. V tomto štádiu je systém aktívny a sleduje pohyb vo vyznačených oblastiach. V ľavom dolnom rohu je informácia o tom, či je zapnutý časozber (nastavenie v prvej fázy konfigurácie). Tlačidlo *nahrávaj* na zapnutie a aj následne vypnutie videozáznamu celého obrazu webkamery v predvolenom rozlíšení a snímkovaní. Po ukončení nahrávania sa otvorí štandardné systémové okno na uloženie súboru, teda videa. Tlačidlo *off* slúži na vypnutie režimu behu a návrat do druhej fázy konfigurácie.

# **4. Závislosti**

## 4.1. Externé rozhrania

4.1.1. OpenCv

Opencv je externá knižnica hlavne na spracovanie obrazu, ktorú systém využíva. V systéme sú použité funkcie z tejto knižnice, ktoré zabezpečia:

* získanie obrazu z webkamery
* konverziu obrazu z RGB kódovania na čiernobiele a naopak
* zmenu rozmerov obrazu
* rozdiel dvoch obrazov ako nový obraz
* získanie časti obrazu
* kreslenie jednoduchých tvarov na obraz

4.1.2. NumPy

NumPy je externá knižnica obsahujúca základné vedecké výpočtové funkcie ako napríklad: prácu s N-dimensionálnymi objektami, maticové operácie a podobne. OpenCV využíva matice z NumPy ako pixelovú reprezentáciu obrazu.

## 4.2. Perzistentné súbory

4.2.1. Frame

* Frame - jeden obraz zaznamenaný kamerou, bude uchovávaný v troj-rozmernej matici kde dva rozmery sú rozlíšenie obrazu a tretí rozmer je počet farebných kanálov teda 3, ak bude použitá metóda RGB na ukladanie farieb pri farebnom obraze, alebo 1 ak sa bude ukladať čiernobiely obraz.

4.2.2. Záznam pohybovej udalosti

* Zaznamenané videá pohybovej udalosti. Tieto záznamy budú uložené vo formáte .MP4 v používateľom zvolenom priečinku. Zmena tohto priečinku je možná v konfigurácií
* Tvar názvu súboru bude:

<timestamp>#<číslo udalosti, ktorá nastala počas tohto behu programu>

4.2.3 Časozberný záznam

* Zaznamenané časozberné videá. Tieto záznamy budú uložené vo formáte .MP4 v používateľom zvolenom priečinku. Zmena tohto priečinku je možná v konfigurácií
* Tvar názvu súboru bude:

<timestamp začiatku nahrávania>#<timestamp konca nahrávania>

4.2.4 Záznam log

* Log - súbory. Bude to textovy dokument .TXT s nasledujúcim formátom obsahu:

<timestamp>\_<dĺžka trvania pohybovej udalosti v sekundách>\_<toString oblasti v ktorej bol pohyb zaznamenaný>#<číslo udalosti, ktorá nastala počas tohto behu programu>

príklad: 2018-10-30 14:08:49.050215\_10.5\_RegionOfInterest(start=Point(X=0, Y=0), end=Point(X=0, Y=0), ignored\_areas=[], sensitivity=10)#1

* Ku každému záznamu bude prislúchať jeden obrázok formátu .JPG v priečinku logPictures. Názov obrázku bude obsahovať timestamp a poradové číslo v tvare

<timestamp>#<číslo udalosti, ktorá nastala počas tohto behu programu>

* Logovanie bude zabezpečuje python class Logger

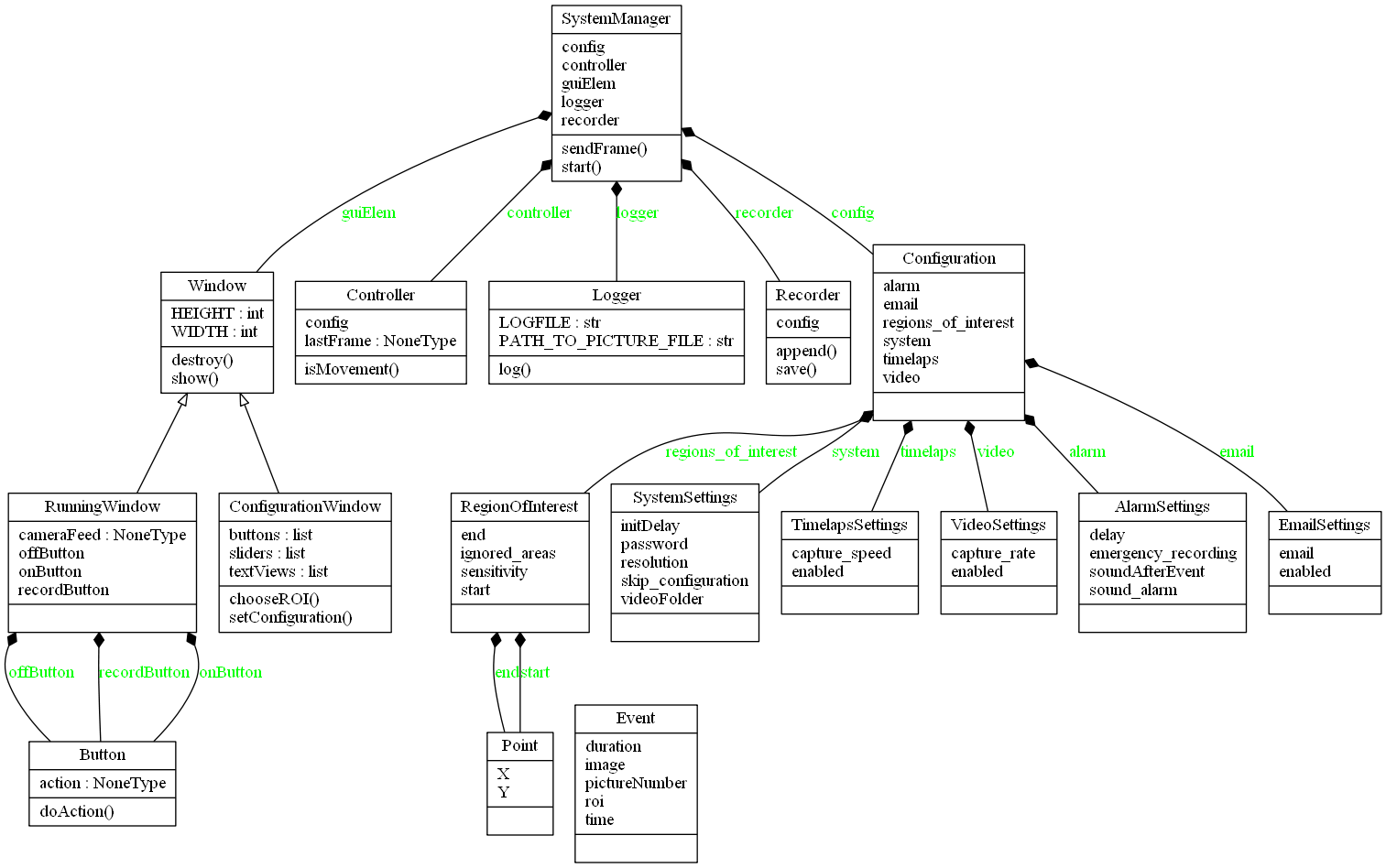
4.2.5 Konfiguračný súbor

* Konfiguračný súbor. Na ukladanie konfigurácie využijeme python modul pickle, ktorým serializujeme class Configuration, ktorá obsahuje všetky konfigurovateľné nastavenia systému.

# **5. Implementácia:**

## 5.1. Triedny diagram

V tejto časti uvádzame detailný triedny diagram. Ten vychádza z komponentného diagramu, pričom niektoré komponenty sa skladajú z viacerých tried. Diagram je vytvorený pomocou nástroja Pyreverse, ktorý je súčasťou balíka Pylint (<https://pypi.org/project/pylint/>)



*diagram 2: triedny diagram*

## 5.2. Popis tried

Na vrchu hierarchie (vizuálne v diagrame, ale aj logicky v systéme) je trieda *SystemManager,* čo zodpovedá komponentu a procesu Riadenie snímania (viď. kapitola 2. Rozdelenie). Spoločne s ďalšími triedami v systéme ako *Recorder, Logger, Controller,* či *Configuration* bude reálne v systéme počas behu iba jedna inštancia/objekt týchto tried, takže ich možno implementovať návrhovým vzorom *Singleton.*

Úlohou SystemManagera bude rozdávať úlohy ostatným častiam systému. SystemManager spustí hlavný cyklus v ktorom beží načítanie obrazu z kamery, ktorý následné posiela na ďalšie spracovanie ostatným častiam.

Controller je objekt, ktorého hlavnou úlohou je rozpoznať pohyb na videu a oznámiť to SystemManagerovi

Logger a Recorder sa starajú o ukladanie údajov. Logger ukladá základné informácie o pohybovej udalosti ktorá nastala do log filu, tiež ukladá fotografiu z udalosti. Recorder zaznamenáva a ukladá video ak užívateľ zvolil možnosť nahrávania.

RunningWindow je GUI element zobrazujúci systém počas snímania pohybu v oblastiach záujmu.

ConfigurationWindow je GIU element, ktorý umožní užívateľovi zmeniť konfiguračné nastavenia systému. Je spustený hneď po štarte aplikácie ak si používateľ nezmenil konfiguráciu inak.

Riadenie systému bude úplne oddelené od grafického rozhrania, ktoré bude bežať samostatna a iba reagovať na vstupy užívateľa (kliknutie na tlačidlo, posunutie bežca a pod.), tým, že o tom uvedomí *SystemManagera*, ktorý podnet náležite spracuje.

Diagram bol generovaný zo zdrojového kódu použitím modulu PyLint, ktorý obsahuje modul PyReverse nasledovným príkazom

C:\git\visionlabfotopasca> pyreverse -AS -o png Src

význam - prepínač A - zobrazia sa všetky nadtriedy triedy

* prepínač S - všetky asociatívne závislosti sa zobrazia aj rekurzívne
* prepínač o - definuje výstupný formát súboru
* Src je názov hlavného package, z ktorého tried generujeme diagram

## 

## 

## 

## 5.3. Algoritmus spracovania obrazu

**def** isMovement(self, frame) -> Dict[int: bool] :

*## mapa -> kluc je roi -> hodnota Bool (true ak nastal pohyb, inak false )*

res = dict()

*# konvezria na gray scale img*

gray\_img = gray\_frame = cv2.cvtColor(frame, cv2.COLOR\_BGR2GRAY)

*# ak uz boli dva framy a teda mame frame na porovnanie*

**if** self.lastFrame **is not None**:

*# spravime rozdiel pixelov v obrazkoch*

diff = cv2.absdiff(self.lastFrame, gray\_frame)

**for** rio **in** self.config.regions\_of\_interest:

*# vybereme oblast roi*

rio\_img = diff[rio.start.Y:rio.end.Y, rio.start.Y:rio.end.Y]

*# spravime treshlold podla sensitivity*

thresh = cv2.threshold(rio\_img,rio.sensitivity,255,cv2.THRESH\_BINARY)

*# percentualna zmena*

percentage\_change = (cv2.sumElems(thresh)/255)\

/ ((rio.end.Y - rio.start.Y)\

\* (rio.end.Y-rio.start.Y)) \* 100

*# aka zmena presiahla 30 percent nastala udalost co zaznacime*

**if** percentage\_change > 30:

res[rio] = **True**

**else**:

res[rio] = **False**

self.lastFrame = frame

**return** res

## 5.4. Cieľové prostredie

Predpokladáme, že cieľovým prostredím bude počítač so systémom *Windows 10* (kompatibilitu so staršími verziami nezaručujeme), ktorý bude mať pripojenú kompatibilnú webkameru. Systém bude manuálne zapínať a vypínať samotný používateľ. Pri náhlom výpadku prúdu, alebo inej poruche by žiadne dáta softvéru nemali byť ohrozené, teda všetky zaznamenané snímky a logy budú uložené na disku. Ak však prebieha záznam videa (či už spustený manuálne, alebo automaticky pri poplachu), je nutné, aby bol systém korektne ukončený a mal tak príležitosť tento záznam správne uložiť.