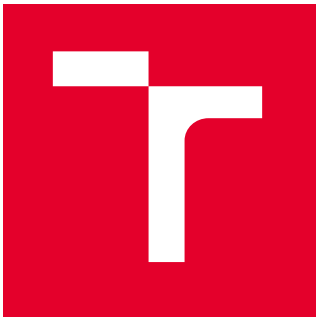


VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS

SYSTÉM MONITOROVÁNÍ IT INFRASTRUKTUR POMOCÍ ROBOTICKÝCH AUTOMATIZOVANÝCH PROCESŮ

IT INFRASTRUCTURE MONITORING SYSTEM USING ROBOTIC PROCESS AUTOMATION

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Matěj Valíček**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Marek Sikora**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Informační bezpečnost**

Ústav telekomunikací

Student: Matěj Valíček ***ID:*** 230903

Ročník: 3 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Systém monitorování IT infrastruktur pomocí robotických automatizovaných procesů

POKYNY PRO VYPRACOVÁNÍ:

Obvyklé firemní či soukromé IT infrastruktury obsahují různé druhy prvků pro zajištění potřebných služeb. Může jít o hardwarové prvky jako záložní zdroje, NAS servery, Routery, Switche, nebo také o softwarové prvky jako např. zálohovací software, antivirové programy, poštovní servery, atd. Ve většině těchto zařízení lze definovat zasílání e-mailových zpráv, ať už jde o pravidelné reporty, nebo hlášení problémů.
Úkolem této bakalářské práce je vytvořit robotizovaný nástroj pro automatizované vyhodnocování příchozích emailových notifikací. Nástroj bude schopen vykonávat třídění či značkování e-mailových notifikací dle předem stanovené logiky v určené e-mailové schránce. Dalším úkolem práce je obohatit nástroj o funkci zasílání notifikací o průběhu, analýzy dochvilnosti, plánování automatického spuštění a export dat o dochvilnosti notifikací. Celé řešení bude navrženo a optimalizováno pro co nejjednodušší obsluhu administrátorem.

DOPORUČENÁ LITERATURA:

Podle pokynů vedoucího práce.

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 26.5.2023

Vedoucí práce: Ing. Marek Sikora
Konzultant: Ing. Jan Sikora (APROSYS ICT s.r.o.)

doc. Ing. Jan Hajný, Ph.D.
předseda rady studijního programu

ABSTRAKT

Cílem této bakalářské práce bylo vytvoření nástroje pro kategorizaci elektronické pošty za pomoci Robotických Automatizačních Procesů (RPA). Kategorizace probíhá na základě zprávy s kategorií „_TEMPLATE“ a podobnosti jejího předmětu s předmětem příchozí zprávy. Robot dále analyzuje dochvilnost zpráv a v případě zpoždění je o této situaci obeznámen administrátor. Na závěr jsou zprávy přesunuty do archivních složek, tak aby v hlavní složce zůstala vždy pouze jedna zpráva od každého předmětu.

KLÍČOVÁ SLOVA

E-mail, automatizace, kategorie, RPA, UiPath, robot, šablona, Outlook, workflow, předmět

ABSTRACT

The goal of this bachelor's thesis was to create a tool for categorizing elektronic mail using Robotic Process Automation (RPA). The categorization is based on a message with category „_TEMPLATE“ and the similarity of its subject with the subject of the incoming message. The robot further analyzes the punctuality of the messages and in case of delay, the administrator is notified about the situation. Finally, messages are moved to the archive folders so that only one message from each subject remains in the main folder at any time.

KEYWORDS

E-mail, automation, category, RPA, UiPath, robot, template, Outlook, workflow, subject

VALÍČEK, Matěj. *SYSTÉM MONITOROVÁNÍ IT INFRASTRUKTUR POMOCÍ ROBOTICKÝCH AUTOMATIZOVANÝCH PROCESŮ*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2022, 87 s. Bakalářská práce. Vedoucí práce: Ing. Marek Sikora

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Matěj Valíček
VUT ID autora:	230903
Typ práce:	Bakalářská práce
Akademický rok:	2022/23
Téma závěrečné práce:	SYSTÉM MONITOROVÁNÍ IT INFRASTRUKTUR POMOCÍ ROBOTICKÝCH AUTOMATIZOVANÝCH PROCESŮ

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Marku Sikorovi za konzultace, trpělivost a poskytnutí testovacích dat k praktické části.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Tato bakalářská práce se zabývá automatizací s využitím robotických procesů pro jednodušší správu zařízení v síti. Na většině prvků v sítí nebo softwaru lze nastavit zasílání hlášení, ať už pravidelných či mimořádných. Tato hlášení jsou zcela jistě užitečná, ale v případě většího počtu prvků mají za následek zahlcení schránky administrátora. Praktickou částí této práce je proto vytvoření robota pro kategorizaci poštovní schránky podle předem definovaných šablon.

Mezi cíle práce patří seznámení čtenáře s automatizacemi procesů různého typu a následně zaměření na automatizace robotickými procesy. Hlavním bodem je vytvoření robota, který využije přihlášenou aplikaci `Outlook`, roztřídí e-maily ve vybraných složkách a poté přesune starou poštu do archivu.

Kód robota je napsán ve vývojovém prostředí `UiPath Studio` s implementovaným jazykem `Visual Basic`. Bot pracuje s e-maily, které mají kategorii „_TEMPLATE“. Tyto zprávy jsou uloženy do mezi-paměti a nově příchozí zprávy jsou s nimi poté porovnávány. V případě shody je označen druhou kategorií u šablony, která identifikuje typ příchozího hlášení ze zařízení. Shoda nastává, pokud se předmět příchozího mailu rovná předmětu šablony.

Dalším krokem je kontrola dochvilnosti zpráv, ty jsou mezi sebou porovnávány. Pokud vzniklo zpozdění větší, než je dovoleno, je o této situaci informován administrátor. V neposlední řadě jsou zprávy přesouvány do archivních složek podle příchozího data v rámci jednotlivých předmětů.

1 Počítačová sít

Počítačová síť se dá definovat jako seskupení více počítačových stanic nebo jiných prvků, které si mezi sebou dokáží vyměňovat informace. Komunikace je zprostředkována protokoly TCP/IP (*Transmission Control Protocol/Internet Protocol*). Mezi další prvky patří např. směrovač, mobilní telefon nebo tiskárna [1].

1.1 ISO/OSI

Pro pochopení funkcionality sady protokolů TCP/IP je nutné znát základní síťový model ISO/OSI *(International Standards Organisation / Open System Interconnection*). Model je sestaven ze sedmi vrstev, tak jak lze vidět na obrázku 1.1. Každá vrstva poskytuje své služby vrstvě vyšší, takže např. fyzická vrstva poskytuje své služby vrstvě linkové (spojové).

Obr. 1.1: Vrstvový model ISO/OSI

1.1.1 Shrnutí funkcí jednotlivých vrstev ISO/OSI

Fyzická vrstva

Je to nejnižší vrstva modelu ISO/OSI. Jak již z názvu vyplývá, tato vrstva se zabývá fyzickými prostředky sítě, např. kabelovým spojením, napěťovými úrovni signálu a reprezentací dat. Na úrovni této vrstvy data reprezentuje jednotka `bit`.

Spojová vrstva

Spojová, nebo též linková vrstva, využívá služeb vrstvy fyzické pro přenos bloků dat, které se nazývají rámce. Jako další funkce lze uvést identifikaci těchto rámců – začátek a konec, detekce i korekce chyb v přenosu a ochrana proti zahlcení.

Síťová vrstva

Funkce síťové vrstvy se dá shrnout jako logické adresování a směrování sítí, které spolu fyzicky nesousedí. Na této vrstvě se definují tzv. IP adresy, které slouží pro identifikaci stanice, s kterou chce uživatel komunikovat. Data jsou posílána ve formě paketu.

Transportní vrstva

Hlavním úkolem této vrstvy je adresování určitého aplikačního protokolu. Segmentace na straně odesilatele, zpětné skládání dat dohromady u příjemce, řízení spojení a toku dat.

Relační vrstva

Relační vrstva má za úkol navázání, udržení a následné ukončení relace. Dále se stará o přepojování uzlů v případě nedostupnosti nebo synchronizaci spojení

Prezentační vrstva

V rámci počítačové sítě může uživatel komunikovat se stanicí s jiným operačním systémem nebo typem kódování. Je proto nutné zajistit převod těchto dat. K tomu slouží prezentační vrstva, která má na starosti kódování znaků, šifrování a kompresi.

Aplikační vrstva

Nejvyšší vrstva má za úkol zpřístupňovat všechny vrstvy aplikacím a dohlížet na jejich spojení. [2] [4] [5]

1.2 TCP/IP

Hlavním rozdílem oproti ISO/OSI je v počtu vrstev na kterých pracuje. TCP/IP totiž funguje pouze na čtyřech - **Vrstva síťového rozhraní, Síťová vrstva, Transportní vrstva** a **Aplikační vrstva** (obrázek 1.2). Dalším rozdílem je v možnostech implementace.

Obr. 1.2: Vrstvy TCP/IP

Vrstva síťového rozhraní

Nejnižší vrstva vznikla spojením fyzické a spojové vrstvy. Má na starosti přístup k fyzickým prostředků, posílání a přijímání bloků dat. Dále není specifikována, je závislá na přenosové technologii.

Síťová vrstva

Tato vrstva funkčně odpovídá vrstvě v modelu ISO/OSI. Nazývána pomocí protokolu, kterým je realizována - IPv4 (IP verze 4) nebo IPv6 (IP verze 6). Úkolem vrstvy je dohled nad posílanými pakety, dost často prostřednictvím dalším sítě. K tomu slouží zařízení jménem směrovač (anglicky *router*).

Transportní vrstva

Hlavní funkcí této vrstvy je zajistit spolehlivý přenos a možnost měnit nespojovaný přenos na spojovaný mezi dvěma aplikačními programy. K tomu slouží protokol TCP (*Transmission Cotrol Protocol*). Dalším typem přenosu je pomocí protokolu UDP (*User Datagram Protocol*), ten na rozdíl od TCP nezajišťuje spolehlivost přenosu.

Aplikační vrstva

Jakožto nejvyšší vrstva poskytuje jednotlivé aplikační protokoly přímo nižší vrstvě, tedy transportní. Relační a prezentační vrstvy nejsou přítomny a pokud aplikace

tyto služby potřebují, zajistí si jej sami. Na této vrstvě je dále definováno mnoho protokolů jako např. SMTP (Simple Mail Transfer Protocol), HTTP (Hypertext Transfer Protocol) nebo FTP (File Transfer Protocol). [2] [4] [5]

1.3 Poštovní protokoly

SMTP – Simple Mail Transfer Protocol

Tento protokol je standard pro přenos elektronické pošty Internetem. Využívá protokol TCP, tedy spolehlivý typ přenosu. Server naslouchá na portu **TCP/25**. V dnešní době se používá rozšířená verze protokolu ESMTP (*Extended Simple Mail Transfer Protocol*), ta rozšiřuje původní verzi např. o potvrzování doručení elektronické zprávy. Pro přístup do poštovní schránky se používají protokoly – POP3 a IMAP4. [5]

Bez přítomnosti tohoto protokolu by bylo bezvýznamné automatizovat kategorizaci poštovní schránky, jelikož by přenášení zpráv v síti Internet neexistovalo.

POP3 – Post Office Protocol 3

Protokol POP3 slouží pro jednoduchý a rychlý přístup do elektronické poštovní schránky. Využívá k tomu spolehlivý přenos, server při nezabezpečeném spojení naslouchá na portu **TCP/110**. Zabezpečený provoz je realizován na portu **TCP/995**. Nevýhodou je, že se pošta ukládá lokálně, nikoliv na poštovním serveru. To má za následek stažení veškeré pošty (i tu kterou si uživatel nevyžádal, např. spam).

IMAP4 – Internet Message Access Protocol 4

Protokol IMAP4 slouží pro přístup k elektronické poště. Využívá TCP, tedy spolehlivý přenos. Při nezabezpečeném provozu naslouchá na **TCP/143**, v případě zabezpečeném na **TCP/993**. Na rozdíl od protokolu POP se zprávy ukládají na vzdáleném serveru. Na konečné stanici se ukládají pouze nezbytné informace. Obsah je stažen až po roz-kliknutí dané zprávy, jinak schránka pracuje s tzv. záhlavím zprávy. Protokol umožňuje připojení více klientů zároveň. [7] [8] [9]

1.4 Přístup k elektronické poště

E-mailová schránka může být přístupná pomocí webové aplikace poskytovatele, např. Gmail.com, Seznam.cz, apod. Dalším způsobem je prostřednictvím e-mailového klienta. Typicky program, který komunikuje se vzdáleným poštovním

serverem. Mezi takové klienty patří např.

- Microsoft Outlook,
- Mozzila Thunderbird,
- Evolution (GNOME),
- Apple Mail.

1.5 Prvky sítě

Prvky sítě se mohou lišit v závislosti na velikosti sítě, podmínkách práce a zaměření. Tato část bude věnována prvkům, které se mohou vyskytnout v síti kancelářských prostor.

1.5.1 Aktivní síťové prvky

Jsou to takové prvky, bez kterých by síť nemohla fungovat. Podílejí se např. na zesilování a úpravy signálu nebo směrování dat.

Mezi takové prvky patří například

- rozbočovač (anglicky *Hub*),
- směrovač (anglicky *Router*),
- přepínač (anglicky *Switch*),
- opakovač (anglicky *Repeater*),
- přístupový bod (anglicky *Access Points*) [10].

Směrovač

Zařízení které funguje na síťové vrstvě a provádí tzv. „směrování“. To je proces rozhodování o cestě paketu v síti. Cesty si poté ukládá do směrovacích tabulek, které poté využívá k rychlejšímu určitění optimální cesty. Používá se v sítích WAN [1] a také pro přijení/oddělení lokální sítě (LAN)[2].

Přepínač

Přepínače pracují na druhé vrstvě modelu ISO/OSI, tedy spojové. Úkolem tohoto zařízení je zesílení přijatého signálu a propojení dvou či více portů. Využívají k tomu tabulky s MAC [3] adresy připojených zařízení.

[1] WAN - World Area Network
[2] LAN - Local Arean Network
[3] MAC - Media Access Control

Opakovač

Jak již z názvu vyplývá úkol tohoto zařízení je zesílení/úprava přijatého signálu. Pracuje na nejnižší vrstvě, tedy fyzické. Hlavním důvodem použití opakovače jsou omezené fyzikální vlastnosti použitého média.

1.5.2 Pasivní síťové prvky

Nejčastěji do této kategorie zařadíme takové prvky, které jsou spíše prostředky pro síťovou komunikaci.

- Přenosové média – Optické vlákno, kroucená dvojlinka, koaxiální kabely nebo volný prostor(bezdrátový přenos).
- Zásuvky a spojky. [11]

1.5.3 Další síťové prvky

To jsou takové prvky, které se nepodílejí na chodu komunikace v síti a slouží pro koncového uživatele. Zařízení z této kategorie je opravdu mnoho, proto zde budou vyjmenovány pouze některé, nejčastější.

- Hardwarové prvky:
  - síťové úložiště,
  - tiskárny,
  - kamerové systémy.
- Softwarové prvky:
  - zálohové systémy,
  - antivirové programy,
  - poštovní servery.

Na většině těchto zařízení, s výjimkou pasivních prvků, lze nastavit pravidelné zasílání hlášení o svém stavu, či výjimečných situací (např. chyby). Tyto hlášení jsou zpravidla zasílány ve formě e-mailu a jsou velice dobrým ukazatelem správného chodu sítě. Ve větších sítí však vzniká potenciální problém se zahlcením poštovní schránky z důvodu velkého počtu zpráv.

Tento problém lze vyřešit automatizací tohoto procesu, tzv. systémem pro automatickou kategorizaci poštovní schránky podle předem určených pravidel v rámci zařízení.

2 Automatizace

S přibývajícími technologiemi je logický krok přesunout práci z člověka na stroj. K tomu slouží automatizace. Ta označuje využívání systému, který nepotřebuje být obsluhován. Výhodou je zmenšení nákladů, větší přesnost a především ušetření lidské práce.

Pro správné pochopení automatizací je nutné definovat dva termíny, které se v této oblasti velmi často používají.

Proces

Tento termín označuje posloupnost kroků a rozhodnutí. Procesy jsou každodenní součástí našich životů. Mezi takové může patřit například:

- příprava jídla,
- uklízení,
- vyzvednutí objednávky,
- výměna žárovky.

Každý z těchto procesů je jasně definován - začátek, konec a hlavně kroky, které se musí vykonat pro jeho splnění. Automatizace je tak efektivní jako její proces, proto nemá smysl automatizovat proces, který není dobře vyladěný. [12]

Robot

Robot je stroj, který dokáže vykonat práci bez nutnosti lidského vstupu. Je součástí vědního oboru „robotika“, která se zabývá vývojem robotů a jejich použití v oblasti automatizace. V současné době existuje mnoho druhů robotů, mezi které patří např.

- předprogramovaný robot,
- lidský robot (anglicky *Humanoid*),
- autonomní robot,
- **softwarový robot**.

Předprogramovaný robot

Tento typ robotů pracuje v hlídaném prostředí, nejčastěji provádí jednoduché a fyzicky náročné práce, které by člověk nebyl schopný vykonat sám. Příkladem takového robota může být mechanické rameno pro skládání motoru vozidla nebo svařování karoserie.

Lidský robot

Dalším typem robota je tzv. *humanoid*. To je robot, který se má blížit lidské podobě. Jeho funkce se blíží těm lidským, dokáže např. běhat, skákat a přenášet předměty. Nejlepší prototypy mají i mimiku obličeje.

Autonomní robot

Autonomní robot pracuje bez nutnosti lidské dohledu. Dokáže pracovat v otevřeném prostředí, jelikož je vybaven senzory, které snímají jeho okolí. Využití těchto robotů je různé, nejběžnějším je např. robotický vysavač.

Softwarový robot

Softwarový robot, nebo též bot, je počítačový program, který vykonává určené úkoly. Nejznámějším softwarovým robotem je tzv. „chatbot“, ten slouží pro jednoduchou komunikaci se zákazníkem. Další využití těchto robotů je např. rozesílání emailů, stahování dat, apod. [13]

2.1 Automatizace výroby

Tento oddíl automatizací se zaměřuje na usnadnění práce v průmyslu, např. v automobilkách. Využívají před-programovaných robotů a multifunkčních strojů, které se dokáží otáčet, zvedat předměty, apod. Díky těmto technologiím je práce rychlejší, přesnější a člověku se zpřístupní jiné (zajímavější) práce, např. obsluha těchto strojů. [14]

2.2 Robotická automatizace procesů

Robotická automatizace procesů, dále RPA, je automatizační software, který využívá boty pro výkon práce za člověka. Softwarový robot vnímá to, co je na obrazovce, dokáže poznat a vyjmout data nebo přepínat mezi aplikacemi. Využívá k tomu uživatelského grafického rozhraní (přesně tak jako člověk). Klíčovou vlastností je omezení chyb člověka a zrychlení prováděného procesu. Robot je totiž přesný, rychlejší a méně nákladný na provoz.

2.2.1 Druhy RPA

Nejzákladnějším rozdělení RPA je podle typu robota. Ty se rozdělují podle toho, jestli je potřeba dohledu člověka nebo nikoliv. Robot, který potřebuje zásah člověka

(ať už na začátku, uprostřed či na konci procesu), se nazývá *Attended* (tzv. obslužné RPA). Druhým typem je *Unatteded* (tzv. bez-obslužné RPA), který tento typ zásahu nevyžaduje.

Attended

Obslužná automatizace, která slouží pro zjednodušení práce uživatele. Nejedná se o celkovou náhradu člověka, ale o pouhé usnadnění složitějších, repetitivních částí. Využívá k tomu akce uživatele jako např, klikání myši, psaní textu a čtení z internetového prohlížeče.

Unatteded

Na rozdíl od předchozího typu RPA je tento typ plně autonomní, tedy nepotřebuje zásah člověka. Práce je zpravidla prováděna na virtuální stanici, kam se robot připojí. Tento typ bota dokáže spouštět aplikace, připojit se k databázi nebo spouštět další roboty pro jejich vzájemnou spolupráci.

2.2.2 Kde se dá RPA využít

Jak již bylo naznačeno v předchozích kapitolách, RPA je možné využít v takovém procesu, kde je potřeba uživatelského vstupu, má jasně definovaná pravidla a často se opakuje. Tento typ automatizace může být využit např.

- ve finančních službách,
- zdravotnictví,
- internetové bezpečnosti (pro kontrolu logů),
- zákaznickém servisu. [15]

2.2.3 Vývojové prostředí

Pro vytvoření bota je nutné mít vývojové prostředí pro RPA. V této oblasti je několik možností, které si zvolit.

- Blueprism,
- UiPath,
- Automation Anywhere.

Blueprism

Blueprism je součástí *Blue Prism Group* a byl založen v roce 2001. Využívá frameworku .NET. Vývojové prostředí je jednoduché na pochopení, používá tzv. *Drag*

and Drop systému, kdy uživatel využívá z předem vytvořených aktivit a pouze je přetahuje do svého pracovního prostředí. [16]

UiPath

Dalším vývojovým prostředím pro RPA je UiPath, který byl založen v roce 2005. Jako jeho konkurence `Blueprism` je postaven na frameworku .NET. Vývojové prostředí je velice intuitivní a přidává možnost objektově orientovaného přístupu. UiPath nabídne i cloudovou službu pro řízení vytvořených robotů a jejich jednoduché spouštění. [17]

Srovnání UiPath X Blueprism

Obě vyjmenované prostředí nabízí předem vytvořené aktivity, které vývojář přetahuje do svého kódu. Mezi takové aktivity samozřejmě patří řídící struktury jako jsou cykly: For each, While, Do while a podmínky. Další aktivity, které jsou speciální právě pro RPA, jsou např. Click, Check App State, Open Application/Browser, Get Outlook Mail Messages, apod. V tabulce 2.2.3 jsou porovnány vybrané funkcionality obou prostředí. [17]

Tab. 2.1: Srovnání funkcionalit vývojových prostředí

Funkcionalita	**UiPath**	**Blueprism**
Založeno na frameworku .NET	✓	✓
Attended mód	✓	✓
Změna kódu v debug módu	×	✓
Objektově orientovaný přístup	✓	✓
Architektura klient-server	×	✓
Kontrola chodu pomocí mobilní aplikace	✓	×
Bezplatné školení vývojářů	✓	×
Možnost spouštět dalšího robota za chodu	×	✓

3 Monitorování sítě

Monitorování sítě je velice důležitým aspektem každé počítačové infrastruktury. Poskytuje neustálé informace o dění v systému, např. zahlcení provozu, vypadnutí linky či kompletní výpadek. Všechny tyto informace jsou předávány správci v reálném čase. Pro tzv. „monitoring“ se používá software, který skenuje danou síť a v případě problému automaticky upozorní pověřenou osobu. Mezi další schopnosti systému patří např:

- Úspory nákladů vynaložené na vyřešení problému v síti (některé systémy pomáhají s analýzou příčin nebo zobrazení vytíženosti síťových prvků).
- Používání neoprávněných aplikací, sledování aktivit uživatelů.
- Včasné varování o bezpečnostních rizicích, nebo kyberneticích útocích. [18]

3.1 Jak monitoring funguje?

Při monitorování sítě se využívá internetových protokolů, jako jsou např. protokol SNMP (*Simple Network Management Protocol*), ICMP (*Internet Control Message Protocol*), nebo Cisco Discovery Protocol v případě využití zařízení Cisco. Důležitým krokem je vytyčení předmětu monitoringu. Mezi takové může patřit např. stav e-mailové sítě, webových serverů nebo konektivity (kontrola aktivních síťových prvků). [18]

SNMP

Využívá se k monitorování a správě sítě. Používá protokol UDP na portech 161 (přijímaní a odesílání zpráv) a 162 (přijímaní trap). Je podporován na široké škále zařízení, např. aktivních síťových prvků, tiskáren a dalších. Pro komunikace vyžaduje dvě strany, první je tzv. správce a druhý agent. Typ komunikace se rozlišuje na dvě situace:

- Správce zasílá dotazy agentovi a přijímá jeho odpovědi.
- Agent zasílá oznámení (neboli trapy) správci. [19] [20]

ICMPv4

Protokol, který se využívá pro kontrolu síťové komunikace. Nepřenáší žádná uživatelská data. Typickým příkladem komunikace typu klient-server. Slouží k předávání zpráv o mimořádných událostí a testování konektivity. ICMP zprávy se klasifikují na 2 skupiny. První je určena k nestandardním stavům, tzv. „error-reporting messages“. Do druhé skupiny spadají zprávy k testování konektivity tzv. „query messages“. [5]

3.1.1 Typy monitorování sítě

Mezi typy monitorování sítě patří:

- analýza síťových paketů – kontrola správného směrování, nebo uživatelů v síti (jestli nenavštěvují zakázané stránky apod),
- sledování aplikací a služeb – zajištění funkčnosti bez omezení,
- kontrola přístupu k síťovým zdrojům. [18]

3.2 Kombinace RPA a monitoringu

I v oblasti monitorování sítě je prostor pro automatizaci lidské práce. Důležitou podmínkou je repetitivní činnost, která je jasně definovaná svými pravidly. Mezi takové může patřit například:

- kontrola stavu zařízení pomocí jejího grafického rozhraní,
- reakce na chybovou hlášku,
- kontrola poštovní schránky a práce s ní,
- spouštění automatických opatření v případě výpadku sítě.

4 Návrh řešení

Tato kapitola bude sloužit k analýze současného stavu a návrhu praktického řešení, které usnadní administrátorovi práci s kategorizací poštovní schránky.

4.1 Současný stav

Za současný stav se dá označit manuální kategorizace zpráv, které administrátor z aktivních síťových prvků přijímá (např. stav, kapacita). Tato vlastnost síťových prvků dokáže být velice užitečná, jelikož administrátor může reagovat na pohotovostní zprávy rychleji, případně s předstihem. Bohužel to může mít i opačný důsledek, kdy je schránka zprávami ze zařízeními zahlcena, jelikož jej administrátor nestíhá kategorizovat. Proto je v jeho zájmu takové situaci předejít.

4.2 Programový přístup

Jako možné předejití situace se zahlcenou poštovní schránkou je využití programového přístupu, které bude tuto práci vykonávat bez nutnosti lidského vstupu. Nástrojů pro tvorbu takového řešení je mnoho, může mezi něj patřit např.:

- programovací jazyky – Python, Java, apod.,
- řešení pomocí RPA – UiPath, Blueprism apod.,
- pravidla v programu `Outlook`.

Jako nástroj pro realizaci programového řešení bylo vybráno RPA, jelikož zde není nutné vytváření vlastních metod a objektů pro práci s programem `Outlook`, ale je možnost využít již vytvořené metody, tzv. aktivity.

4.3 Volba vývojového prostředí

Na trhu je mnoho programů, které nabízí tvorbu RPA. Jedním z nich je právě UiPath, který byl vybrán pro tuto část. Důvodem je například intuitivní nasazení robota na koncovou stanici (viz A), možnost obslužné automatizace a komunitní edici, která je zdarma. Pro detailnější rozebrání a srovnání s dalším prostředí viz tabulka 2.2.3.

4.4 Výhody a nevýhody přístupu RPA

Výhody přístupu RPA se dají přirovnat k výhodám každé jiné automatizace, např.

- ušetření lidské práce,
- zvýšení přesnosti,
- spolehlivost a další.

Nevýhody jsou poté již směřovány přímo na UiPath, mezi takové může patřit:

- nutnost vytvoření účtu u UiPath,
- neschopnost spuštění z příkazové řádky,
- na linuxu nutnost dokerizace.

4.5 Návrh praktické části

Návrh řešení je praktické zamyšlení nad strukturou robota. Dále je vytvořen diagram, podle kterého může být robot vytvořen.
Níže je uvedena jedna z možných struktur robota:

- načtení aktiv z Orchestratoru,
- inicializace konfiguračního souboru,
- zapnutí aplikace `Outlook`,
- čtení a uložení zpráv,
- kategorizace zpráv,
- kontrola dochvilnosti,
- přesunutí zpráv,
- notifikace administrátora.

Načtení aktiv z Orchestratoru

Orchestrator nabízí možnost ukládání hodnot A.3, ať už ve formě klasického textu, nebo přihlašovacích údajů, které jsou šifrované. Mezi potřebné aktiva bude patřit cesta ke konfiguračnímu souboru a přihlašovací údaje do e-mailu.

Inicializace konfiguračního souboru

Robot bude využívat mnoho hodnot, které musí být dynamicky měnitelné, např. názvy složek, které bude robot kategorizovat, účet, apod. Proto je nutné mít soubor, do kterého tyto hodnoty budou uloženy a případně měněny administrátorem. Typem souboru pro uložení těchto informací bude `Excel WorkBook`.

Zapnutí aplikace

Tento krok souvisí s další částí „Čtení a uložení zpráv“ a to z toho důvodu, že aktivitou pro čtení zpráv bude „Get Outlook Mail Message“, která využívá přihlášenou aplikaci `Outlook`. Proto je nezbytné aplikaci zapnout a v případě nutnosti ji přihlásit.

Čtení a uložení zpráv

Přečtené zprávy je nutné filtrovat. Prvním typem zpráv budou takové zprávy, které nemají ani jednu kategorii, tzn. nově příchozí. Dalším typem budou pro šablony, tedy takové zprávy, které mají kategorii „_TEMPLATE“. V neposlední řadě to jsou takové zprávy, které mají jednu kategorii.

Kategorizace zpráv

Kategorizace zpráv bude prováděna na základě podobnosti předmětu zprávy a šablony. V případě shody bude u příchozího e-mailu nastavena druhá kategorie šablony, jenž udává její četnost – např. hodinová, týdenní, měsíční.

Kontrola dochvilnosti

U každého mailu bude zkontrolováno, jestli přišel v požadovanou dobu, respektive v určitém intervalu. K času příchozího mailu se přičte tolerance a poté se rozhodne, jestli se mail opozdil (vzhledem k dalšímu příchozímu mailu).

Přesunutí zpráv

U každého předmětu v dané složce zůstane pouze nejnovější mail a šablona. Všechny ostatní budou přesunuty do archivní pod-složky.

Notifikace administrátora

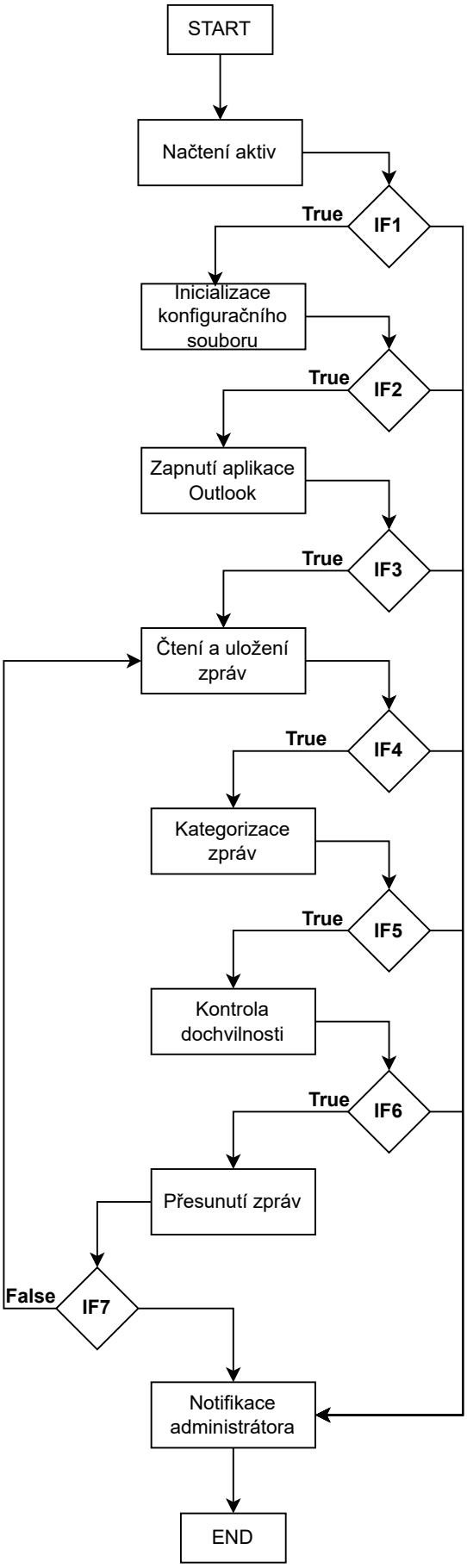
Posledním krokem procesu bude upozornění administrátora na průběh programu. Může to být o úspěšném dokončení, ale taktéž v případě chyby, nulového počtu příchozích zpráv, apod.

4.5.1 Diagram

Pro zpřehlednění slouží diagram kopírující výše uvedené části. Na místech „**IF1**, **IF2**,..“ budou kontrolní podmínky pro případ chyby, např. `Outlook` nebude přihlášený, složka je prázdná, chyba při čtení zpráv, apod. Podmínky jsou vždy popsány na konci každé z částech viz tabulka.

Tab. 4.1: Tabulka s odkazy

Označení podmínky	**Popis v textu (odkaz)**
IF1	5.3
IF2	5.6
IF3	5.10
IF4	5.13
IF5	5.17
IF6	5.18
IF7	5.20



Obr. 4.1: Návrh řešení

5 Výsledky studentské práce

Praktická část této práce je vytvoření bota pro automatickou kategorizaci poštovní schránky, podle předem připravených šablon. Důvodem této automatizace je předejití zahlcení schránky e-maily z koncových zařízení a zároveň možnost kontroly neočekávaných situací na síti, např. chyba při záloze systému.

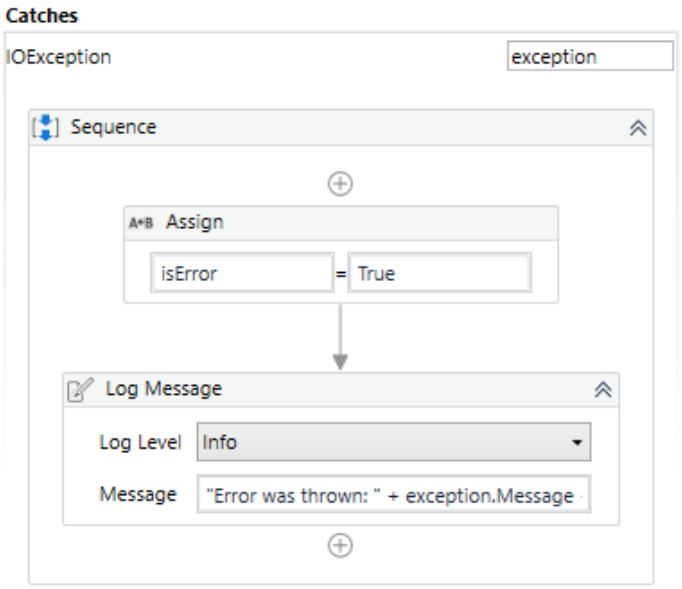
5.1 Programové řešení

Pro vytvoření kódu robota bylo využito tzv. flowchartu, neboli vývojového diagramu. Kód je uvnitř flowchartu rozdělen do 8 logických pod-částí. Každá z nich se stará o specifickou funkci.

- Get Assets (získání aktiv).
- Init Config File (načtení konfiguračního souboru).
- Start Applications (zapnutí aplikací).
- Reading and Saving mails (přečtení a ukládání e-mailů).
- Categorizing mails (kategorizování e-mailů).
- Comparing and Moving mails (porovnávání a přesouvání e-mailů).
- Creating Mail Messages and Log (vytváření zprávy a logu).
- Email Notification (notifikace e-mailem).

Jednotlivé sekvence jsou na sebe navázány a uloženy do tzv. „Try/Catch“ bloků. Tyto bloky mají za úkol zachytit potencionální výjimku (anglicky *exception*). Tato aktivita je složena ze tří částí – **Try**, **Catch** a **Finally**. V části **Try** je uložen kód, který může vyvolat výjimku. Pokud taková situace nastane tato, výjimka bude zachycena úsekem **Catch**. Kód, který se vykoná vždy, je umístěn do části **Finally**. Např. při chybném přečtení e-mailů bot zareaguje na tuto situaci, nepokračuje dále v procesu a odešle zprávu administrátorovi.

V následujících oddílech budou popsány všechny zmíněné pod-části bota, zejména sekvence aktivit v části **Try**. Kód v bloku **Catch** je vždy stejný viz 5.1. Obsahuje aktivitu „Assign“ pro přiřazení hodnoty proměnné `isError = True` a „Log Messages“ aktivitu, která vypíše do logu informace o výjimce.

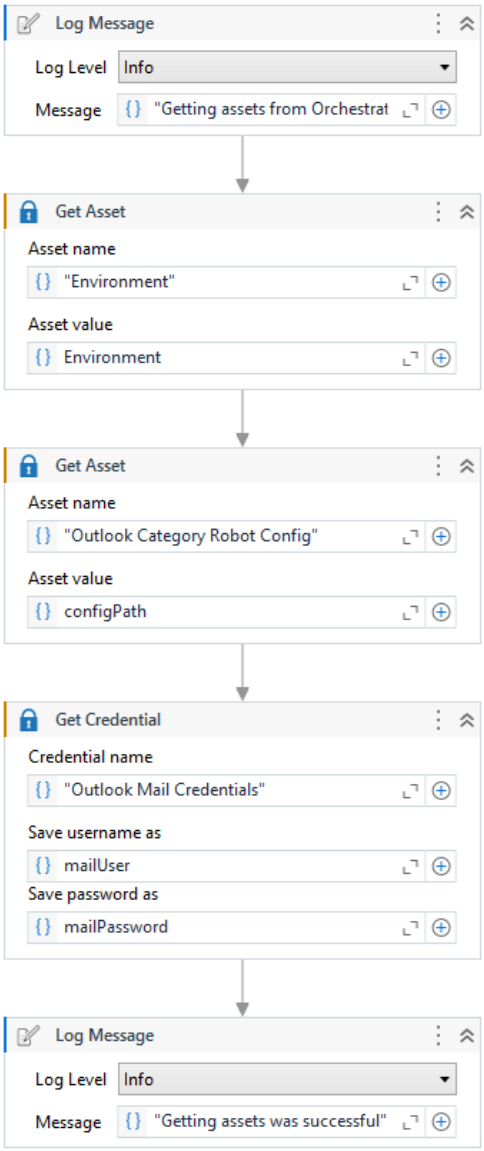


Obr. 5.1: Kód v bloku **Catch**

5.1.1 Get Assets (Získání aktiv)

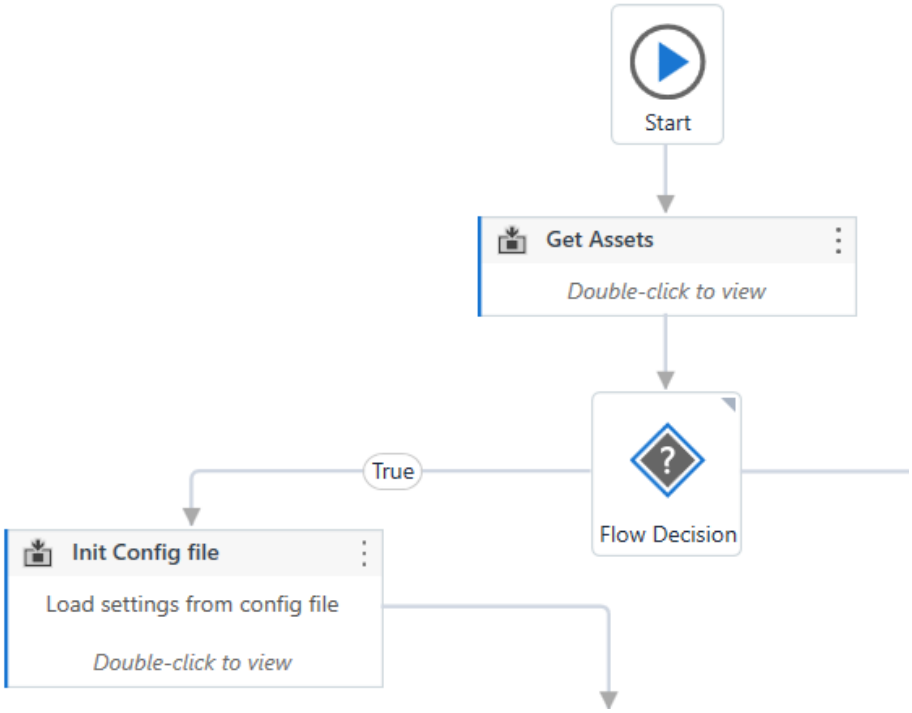
První pod-částí je získání aktiv z Orchestratoru. K tomu slouží předem uložené hodnoty, které robot dále využívá (jak přidat akitiva viz A.3).

Začáteční aktivitou, tak jak lze vidět na obrázku 5.2, je vypsání informace o získávání aktiv do logu. Další tři aktivity - „Get Asset“ slouží pro samotné získání hodnoty z webového cloudu. Vstupním parametrem této aktivity je název aktiva a poté proměnná, do které bude její hodnota uložena. První „Get Asset“ slouží pro získání prostředí, v kterém robot pracuje, např. Produkce. Druhým a důležitějším aktivem je cesta ke konfiguračnímu souboru, ze kterého bot čte další hodnoty, např. e-mailovou schránku, kterou bude procházet. Poslední hodnoty které robot využívá jsou přihlašovací údaje do Outlooku. Na závěr jsou do logu vypsány informace o úspěšném získáním aktiv.



Obr. 5.2: Sekvence aktivit v pod-části Get Assets

Po této pod-části následuje kontrolní podmínka pro případ chyby viz 5.3. Pokud by proměnná `isError` byla nastavena jako `True` (Pravda), robot by nepokračoval dále v procesu. Vypsal by informaci o špatném načtení aktiv do logu a ukončil by se.



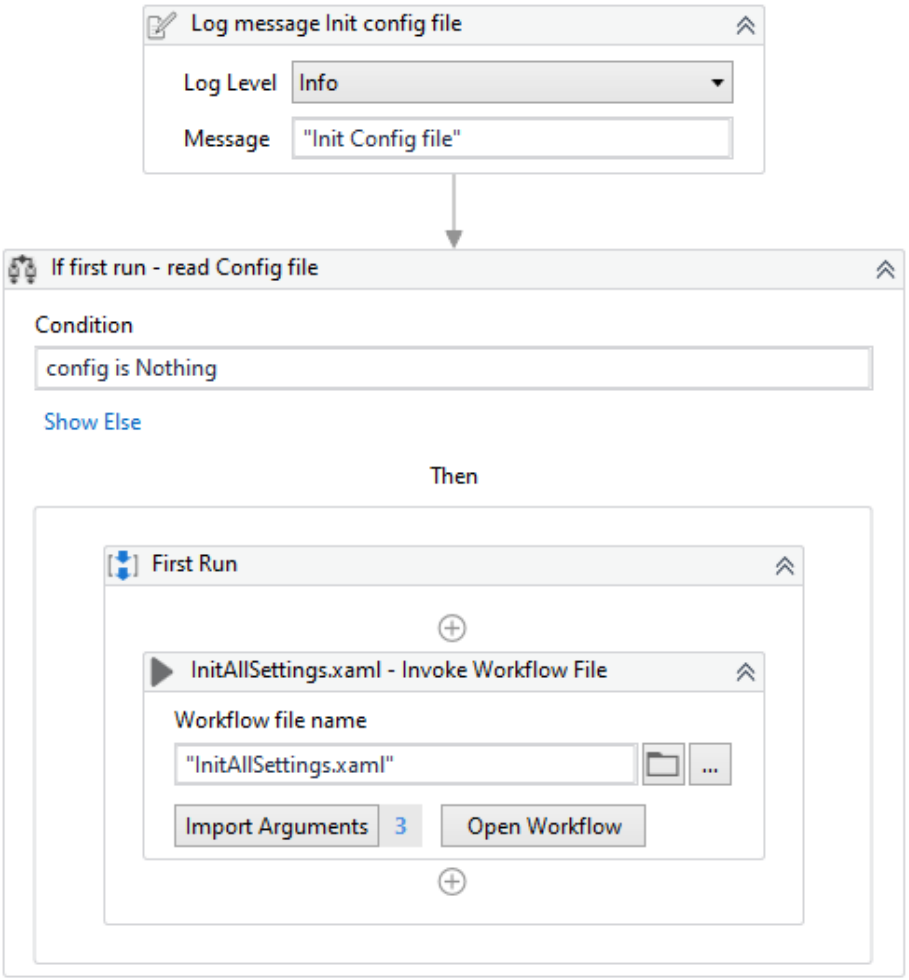
Obr. 5.3: Podmínka mezi dvěma pod-části

5.1.2 Init Config File (Načtení konfiguračního souboru)

Nedílnou součástí robota je načtení konfiguračního souboru. Začátek sekvence je vyobrazen na obrázku 5.4, jako první se vypíše informace do logu. Poté je podmínkou ověřeno, jestli je proměnná `config` prázdná (v případě smyčky by se soubor **načetl** pouze jednou). Při splnění podmínky je využito aktivity „Invoke Workflow file“, jejími vstupnými parametry jsou:

1. název souboru který má být vyvolán,
2. argumenty, které mají být předány.

Výhodou této aktivity je lepší čtení kódu a možnosti změny vstupních argumentů pro dynamické využití daného souboru.



Obr. 5.4: Aktivity v pod-části Init Config File

Workflow – InitAllSettings

Workflow „InitAllSettings“ vyžaduje tři argumenty:

1. cestu ke konfiguračnímu souboru,
2. list (anglicky „sheet“) na kterém se data nacházejí,
3. proměnnou do které budou uloženy.

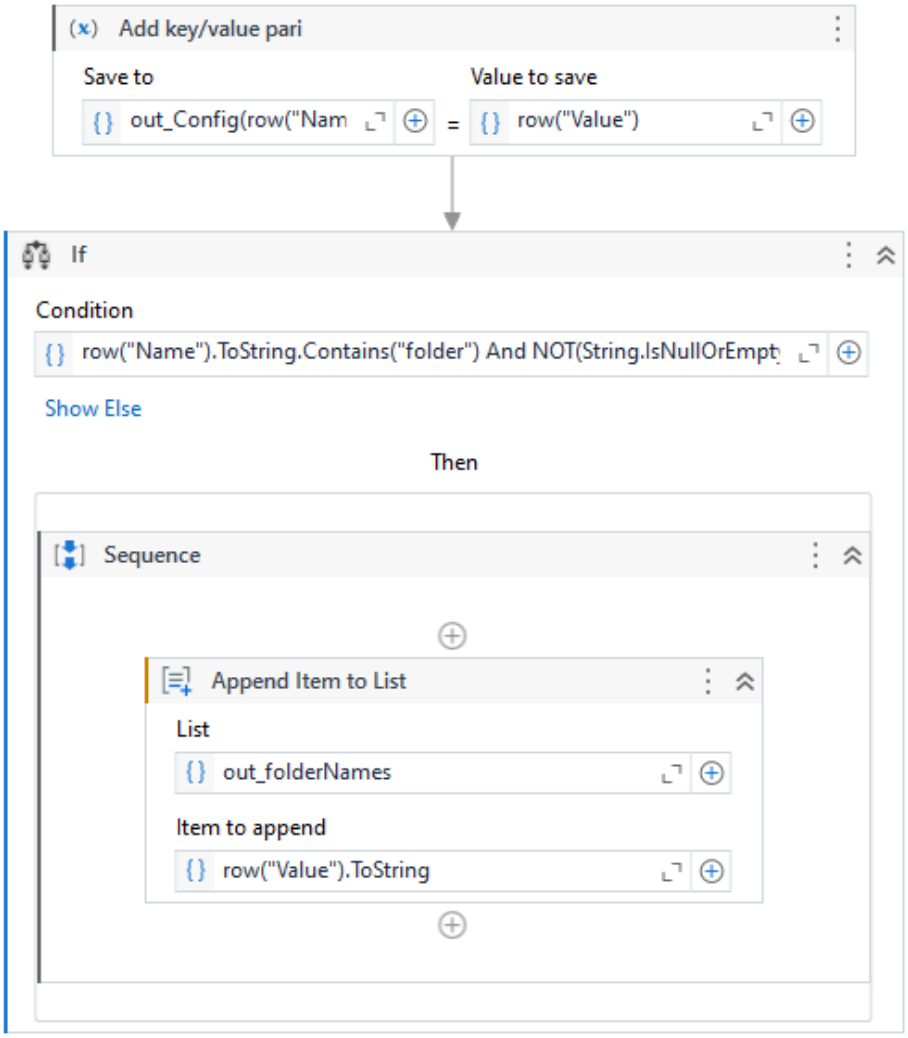
V samotném souboru (obrázek 5.5) je poté pomocí aktivity „Assign“ inicializována výstupní proměnná `out_Config` datového typu `Dictionary` (slovník). Jako klíč slouží jméno řádku (sloupec **Name**, datového typu String) a hodnota je uložena ve formě objektu (sloupec **Value**).

Tab. 5.1: Část konfiguračního souboru

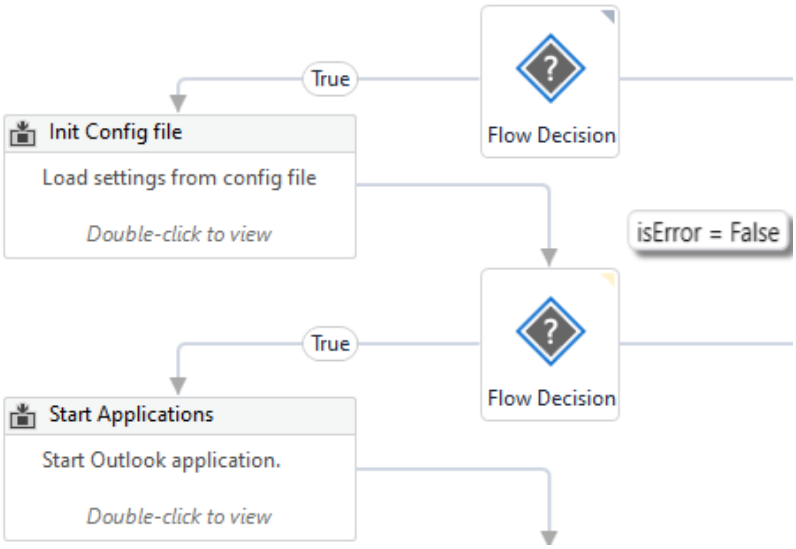
Name	Value	Description
default_path	\\vasemail@gmail.com\	Path to outlook account

Dalším krokem je využití cyklu „For each“. Každý list v tomto souboru se přečte pomocí aktivity „Read range“ se vstupními parametry – **Workbook** path (cesta k Excel workbooku), **SheetName** (název listu) a **Range** (rozshah). Pro cestu a název listu jsou použity argumenty, které vstupují do této workflow a jako rozsah je použit výraz `""`, což má za následek přečtení celého listu. Výstup této aktivity je uložen do proměnné `dtConfig` datového typu `DateTable`.

Pomocí vnořeného „For each“ cyklu, přes všechny řádky v proměnné `dtConfig`, je podmínkou zkontrolováno, zdali tento řádek není prázdný. Pokud není prázdný, tak je tato dvojice klíč/hodnota uložena do výstupní proměnné `out_Config`. Zároveň je zjištěno, jestli se v řádcích „folder“ nachází název složky pro kategorizaci. Pokud ano, tyto názvy se uloží do samostatného listu „**mailFolders**“ , přes který se bude poté iterovat.



Obr. 5.5: Přiřazení dvojice klíč/hodnota a vytvoření listu s názvy složek



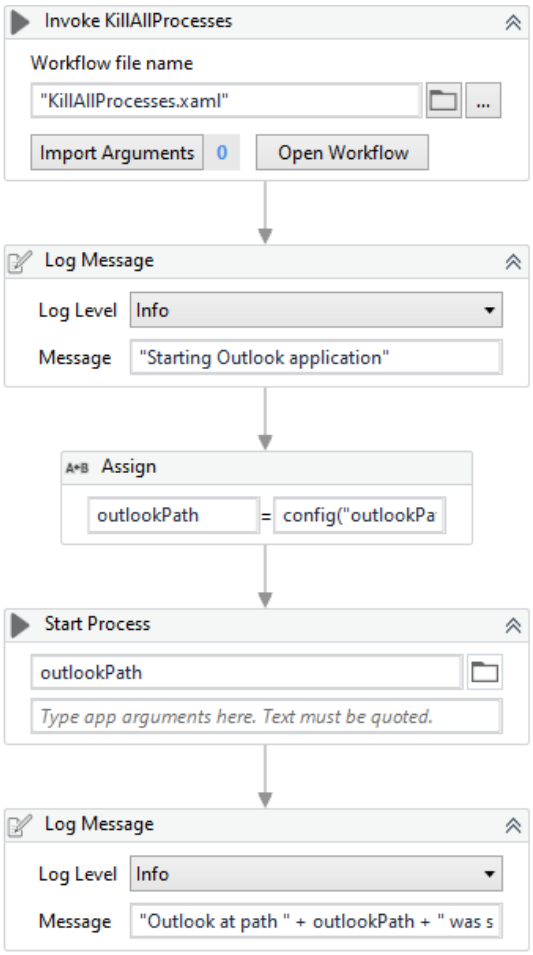
Obr. 5.6: Kontrola proměnné **isError**

Po načtení konfiguračního souboru je vytvořena proměnná datového typu pole (String) „mailMessage“, do nějž se postupně budou přidávat informace o kategorizaci složek. Poté je zkontrolována existence LOG souboru, který je případně smazán, aby byly informace aktuální pro nynější průběh robota. Jako v předešlém kroku i nyní je zkontrolovaná hodnota proměnné `isError`, tak jak jde vidět na obrázku 5.6. V případě `isError = False` robot pokračuje na zapínání aplikací.

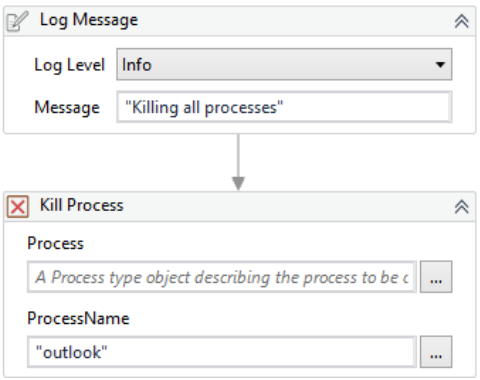
5.1.3 Start Applications (Zapnutí aplikací)

Prvním krokem v této pod-části je vyvolání workflow jménem „KillAllProcesses“ (obrázek 5.7), která ukončí proces jménem „outlook“ viz obrázek 5.8. Toto ukončení procesu je důležité, protože pokud by se robot pokusil zapnout již běžící proces, byla by vyvolána výjimka.

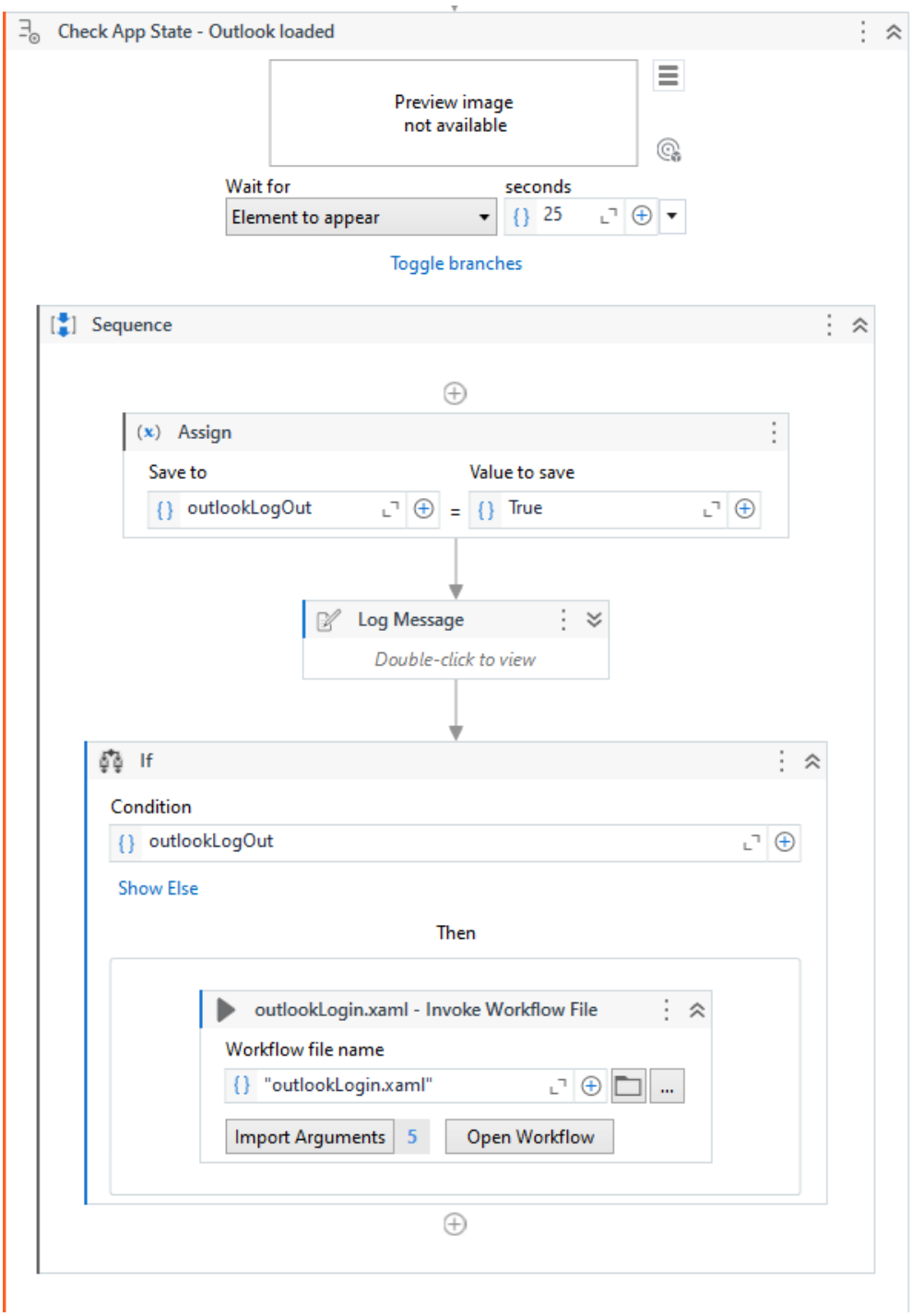
Dále je vypsána informace o zapínání Outlooku do logu a přiřazena hodnota proměnné `OutlookPath` z načteného konfiguračního souboru. Pro zapnutí aplikace `Outlook` je použita aktivita „Start Process“ s cestou k `.exe` souboru jako vstupním parametrem. Poslední aktivitou v této části je „Check App State“, která jde vidět na obrázku 5.9. Slouží k ověření, jestli se daný grafický element nachází na obrazovce. Vrací hodnotu datového typu `Boolean` (True/False). Využívá tzv. selektor, který zachytí grafický prvek jako několik uzlů a atributů. V tomto případě robot sleduje objevení ikony obálky na hlavní obrazovce Outlooku. Jestli robot rozhodne, že se tento prvek na obrazovce nenachází, tak je bráno, že je `Outlook` odhlášený. Proměnná `outlookLogOut` je nastavena jako `True`, informace je vypsána do logu. Pokud je `Outlook` odhlášený je vyvolána workflow jménem „outlookLogin“.



Obr. 5.7: Sekvence aktivit pro zapnutí programu Outlook



Obr. 5.8: Workflow – KillAllProcesses



Obr. 5.9: Kontrola výskytu grafického elementu na obrazovce

outlookLogin

Tento soubor vyžaduje pět argumentů:

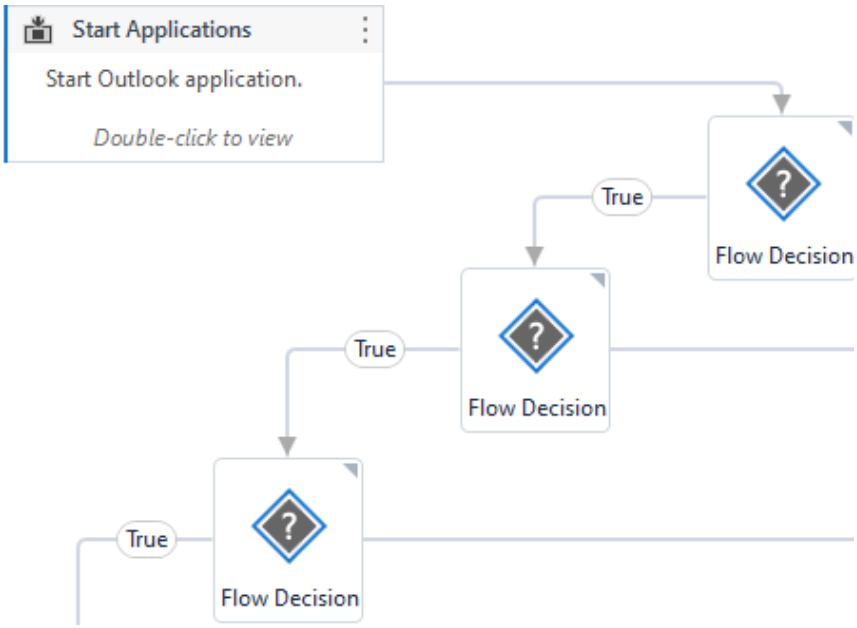
1. proměnnou `outlookLogOut` a proměnnou `outlookLoaded`,
2. přihlašovací jméno a heslo k e-mailu,
3. cestu k Outlooku.

Na začátku souboru se pomocí aktivity „Check App State“ zjistí, jestli je přihlašovací okno viditelné (kroky jsou prováděny 2x – pro český a anglický jazyk, z důvodu různých selektorů, respektive názvu atributů za nimi se skrývající). Pokud je okno viditelné, tak se vyplní přihlašovací údaje (vzaté skrze aktiva z Orchestratoru) a přihlášení se potvrdí. Jako v předešlém kroku je i nyní zkontrolována přítomnost ikony obálky na hlavní straně Outlooku, aby měl robot jistotu, že `Outlook` přihlásil. Pokud ani v této chvíli robot nerozpozná ikonu obálky, je bráno, že je `Outlook` odhlášen, nemůže být přihlášen a proces se ukončí.

Po dokončení pod-části „Starts Application“ následují tři kontrolní podmínky viz 5.10:

1. výskyt chyby (hodnota proměnné `isError`),
2. délka listu „mailFolders“ (jestli existuje složka pro kategorizaci)
3. zdali je `Outlook` přihlášen (hodnota proměnné `outlookLogOut`).

Pokud se v procesu nevyvolala výjimka a ve složce se nachází alespoň 1 e-mail, tak robot pokračuje na kategorizaci těchto zpráv.



Obr. 5.10: Podmínky pro zjištění chyb

5.1.4 Reading and Saving mails (Přečtení a ukládání e-mailů)

V této pod-části budou přečteny všechny zprávy v aktuální složce (názvy brány z listu „mailFolders“ viz 5.1.2. Zprávy, které budou obsahovat kategorii „_TEMPLATE“, budou zvlášť uloženy pro pozdější porovnávání. Další aktivita vyvolá workflow jménem „Read_Mail“.

Read_Mails

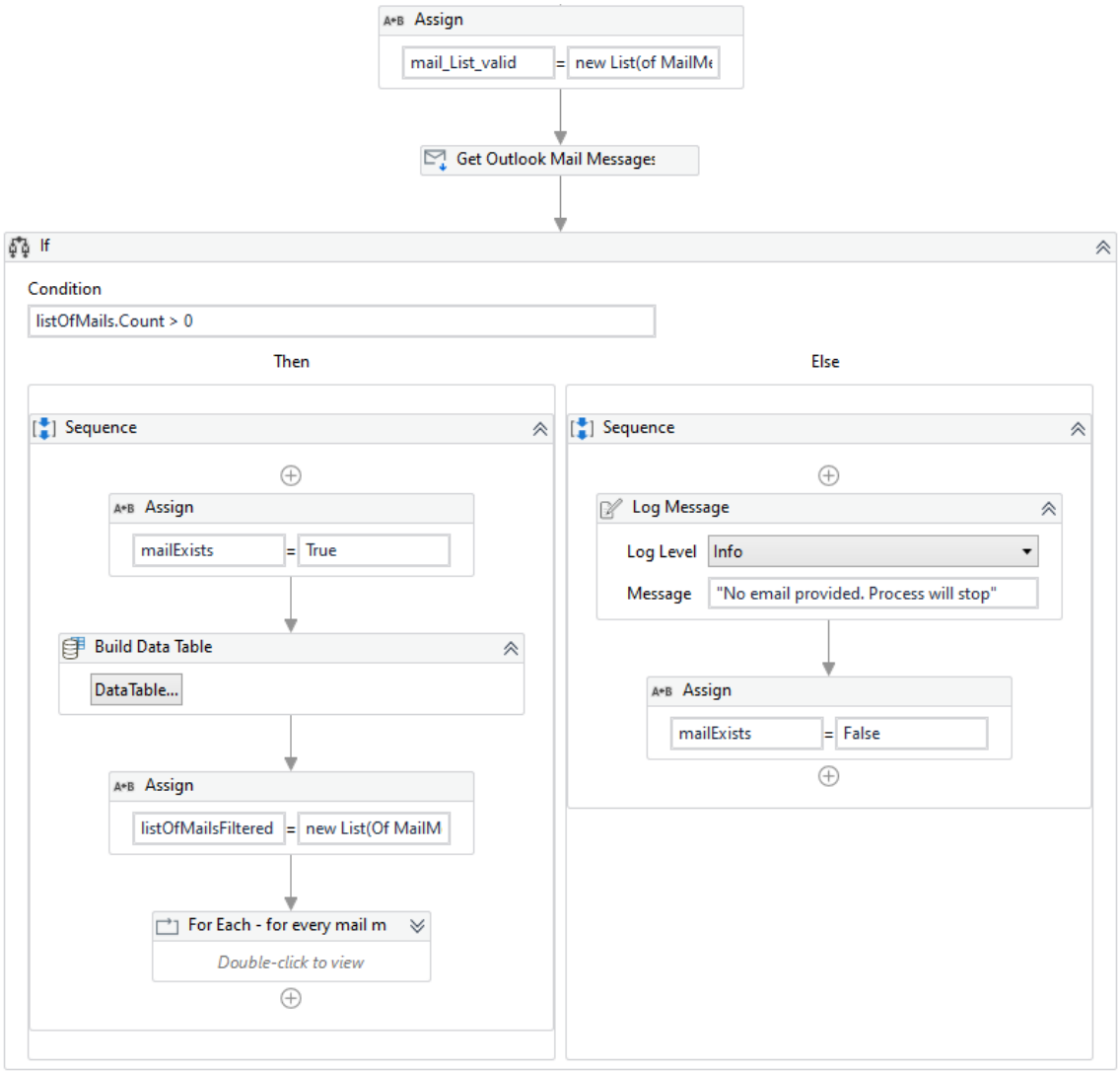
Tento soubor vyžaduje šest argumentů:

1. účet `Outlook`,
2. `Boolean` proměnnou pro stanovení příchodu zpráv `mailExists`,
3. název složky, kterou bude bot procházet,
4. `DataTable` (tabulka) proměnnou pro uložení všech šablon,
5. list pro uložení e-mailů, které se budou kategorizovat (datový typ – `List<MailMessages>`)
6. list pro již kategorizované e-maily (datový typ – `List<MailMessages>`).

Začátkem sekvence je vypsání složky, která se bude kategorizovat, dále inicializace proměnných a samotné přečtení e-mailů pomocí aktivity „Get Outlook Mail Message“. Jako vstupní parametry jsou využity argumenty - účet `Outlook` a složku, kterou bude procházet. Robot čte od nejstarších zpráv a limit je nastaven na 1000 zpráv. Výstup je uložen do pomocné proměnné datového typu `List<MailMessages>`. Podmínkou je zjištěno, jestli byl přečten nějaký e-mail. Pokud ne, robot vypíše informaci do logu a ukončí se. Pro lepší představu slouží obrázek 5.11.V případě nenulového počtu zpráv je pomocí aktivity „Build Data Table“ vytvořena proměnná `dtTemplates` datového typu `DataTable`, do které se budou ukládat vybrané informace šablon. Tyto hodnoty jsou:

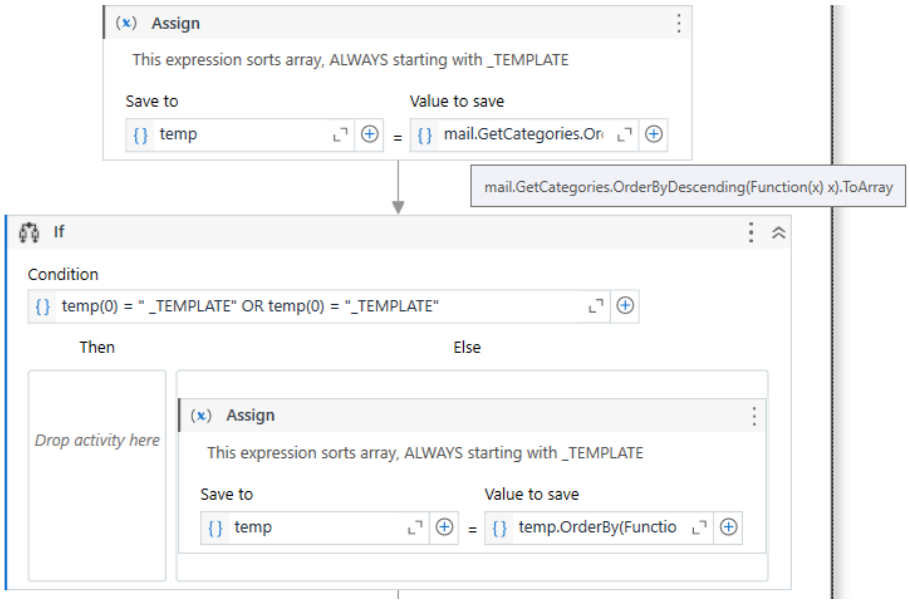
- sender (odesilatel),
- subject (předmět zprávy),
- categories (kategorie, předáváno jako pole hodnot),
- body (tělo e-mailu).

Poté je provedena inicializace proměnné pro ukládání e-mailů.



Obr. 5.11: část workflow Read_Mails

Následující aktivity jsou uvnitř cyklu („For Each“) a aplikují se pro každou příchozí zprávu. První krok je ověření, jestli daný e-mail má nějakou kategorii či nikoliv. Pokud nemá, je to zpráva nově příchozí a je uložena pro kategorizaci. V opačném případě je využito dalšího For cyklu přes všechny kategorie e-mailu a pomocí podmínek zkontrolováno, jestli některá z nich není rovna **„__TEMPLATE“**. Pokud ano, tak se pole s kategoriemi mailu seřadí tak, aby na prvním místě byla kategorie **„__TEMPLATE“** viz obrázek 5.12. Ty jsou poté uloženy do proměnné `dtTemplates`, respektive informace, které jsou vymezeny při vytváření této proměnné. E-maily, které mají pouze jednu kategorii, jsou uloženy do odpovídající proměnné (z důvodu pozdějšího porovnání dle data příchodu).

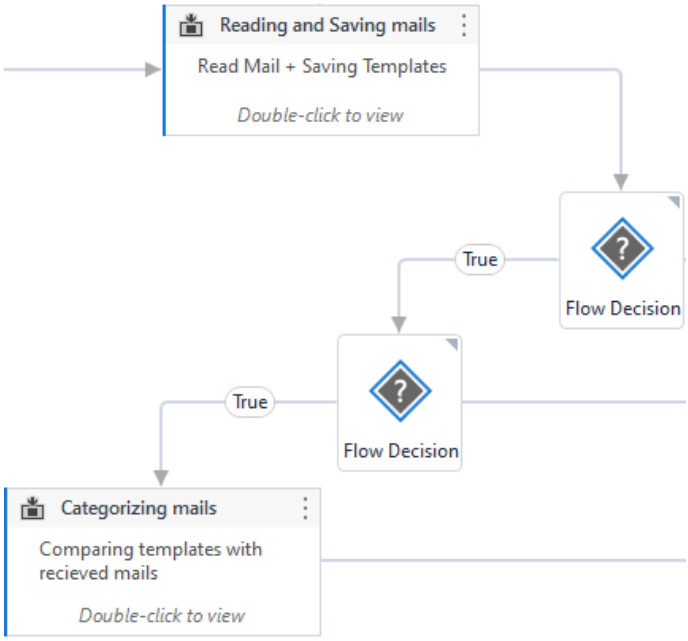


Obr. 5.12: část workflow Read_Mails

Po přečtení a uložení zpráv následují dvě kontrolní podmínky viz 5.13:

1. výskyt chyby (hodnota proměnné `isError`),
2. zdali se ve složce nachází e-mail (hodnota proměnné `mailExists`).

Pokud se v procesu nevyvolala výjimka a ve složce se nachází alespoň 1 e-mail, tak robot pokračuje na kategorizaci těchto zpráv.

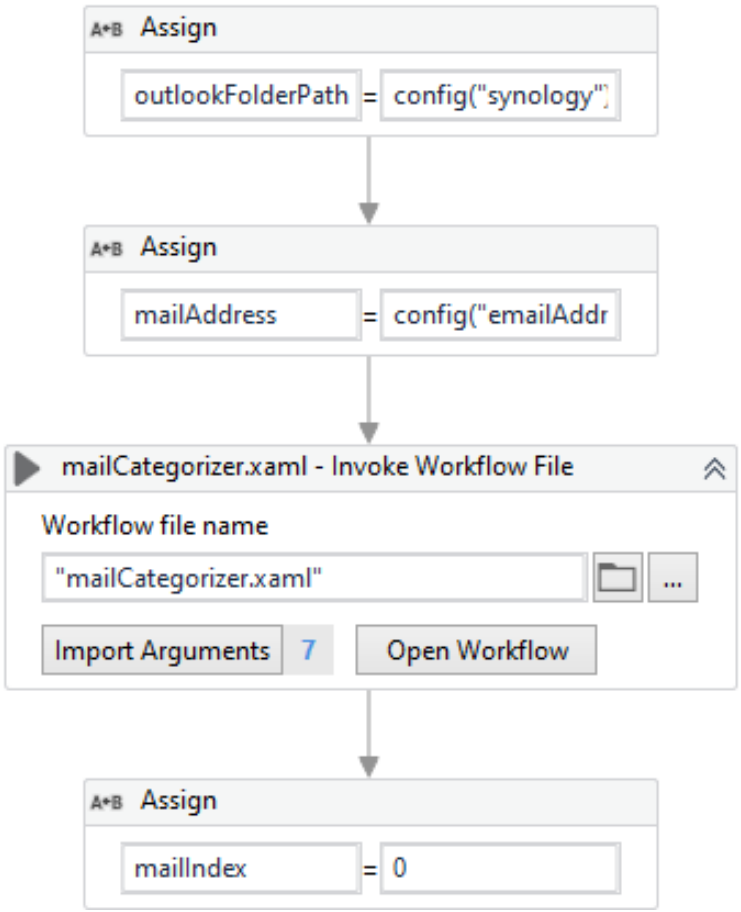


Obr. 5.13: Kontrolní podmínky po přečtení a uložení e-mailů

5.1.5 Categorizing e-mails (Kategorizace e-mailů)

Tato část se zabývá kategorizací zpráv podle šablon (zpráva s kategorií „__**TEMPLATE**“). Zprávy, jejichž předmět se nerovná žádné z šablon, jsou označeny jako nepřečtené pro pozdější určení administrátorem.

Prvním krokem je přiřazení hodnoty proměnným – účet `Outlook` a složka, kterou bude bot kategorizovat. Poté se vyvolá workflow jménem „mailCategorizer“ (obrázek 5.14).



Obr. 5.14: Pod-část pro kategorizaci e-mailů

mailCategorizer

Tato workflow vyžaduje osm argumentů:

1. účet `Outlook`,
2. složku, kterou kategorizuje,
3. proměnnou s uloženými šablonami (`dtTemplates`),
4. list s e-maily,
5. proměnnou pro uložení počtu neznámých zpráv (datového typu `Int32`),
6. proměnnou pro kontrolu indexu zpráv,
7. slovník `mailDates` pro seskupení zpráv při stejném předmětu (klíč je nastaven jako `String` a hodnota `List<MailMessages>`),
8. Boolean proměnnou `noCategoryTemplate` (pokud šablona nemá druhou kategorii).

Tak jak lze vidět na obrázku 5.15, prvním krokem je inicializace proměnných a poté využití For cyklu přes všechny zprávy, které nemají kategorii (`listOfMails_Filtered`). V rámci cyklu je inicializace pomocných proměnných a vstup do aktivity „For Each Row in Data Table“, tedy for cyklu přes všechny řádky proměnné (`dtTemplates`).

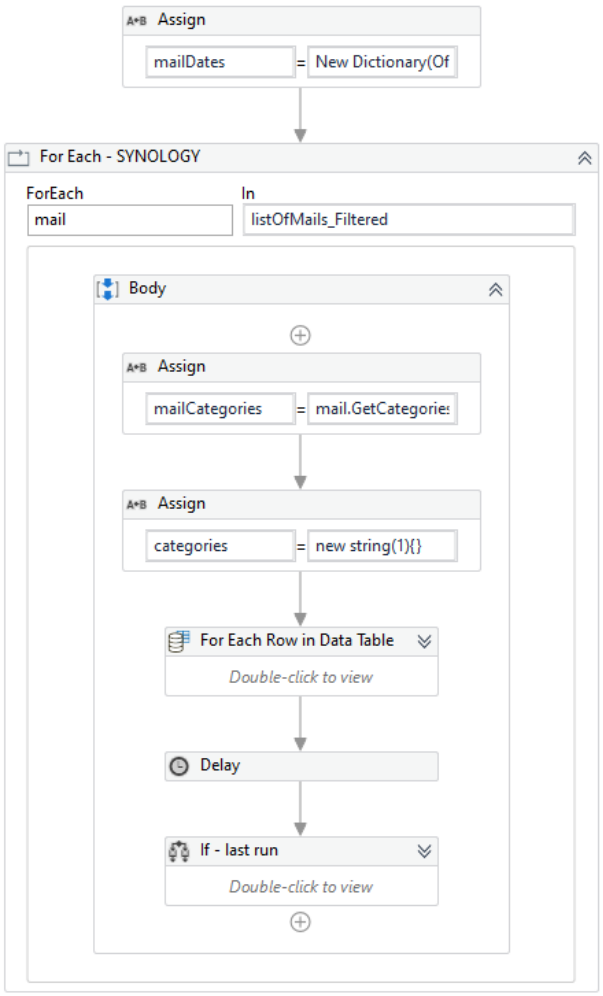
Uvnitř vnořeného cyklu je podmínka pro kontrolu prázdného řetězce (předejití porovnávání s prázdným řádkem v proměnné `dtTemplates`). V případě, kdy řádek není prázdný, je porovnán předmět příchozí zprávy s předmětem šablony. Pokud se nerovnají, nic se nestane a bot pokračuje porovnáváním s dalšími předměty šablon.

Jestliže se předměty rovnají, je vypsána informace o shodě. Aktivitou „Get Row Item“ jsou do pomocné proměnné přiřazeny kategorie šablony. V minulé pod-části „Read Mails“ se kategorie seřadily, tak aby kategorie „__**TEMPLATE**“ byla na prvním místě v seznamu viz 5.12. Proto se pouze ověří, jestli šablona má i druhou kategorii. Pokud ano, tak se pomocí aktivity „Set Outlook Mail Categories“ nastaví u zkoumaného e-mailu. V případě absence další kategorie u šablony je proměnná `noCategoryTemplate` nastavena jako `True` a informace se vypíše do logu.

Dalším krokem v rámci cyklu je uložení této zprávy do slovníku seskupující všechny zprávy stejného předmětu. Tento slovník bude využit v následující pod-části pro přesouvání e-mailů.

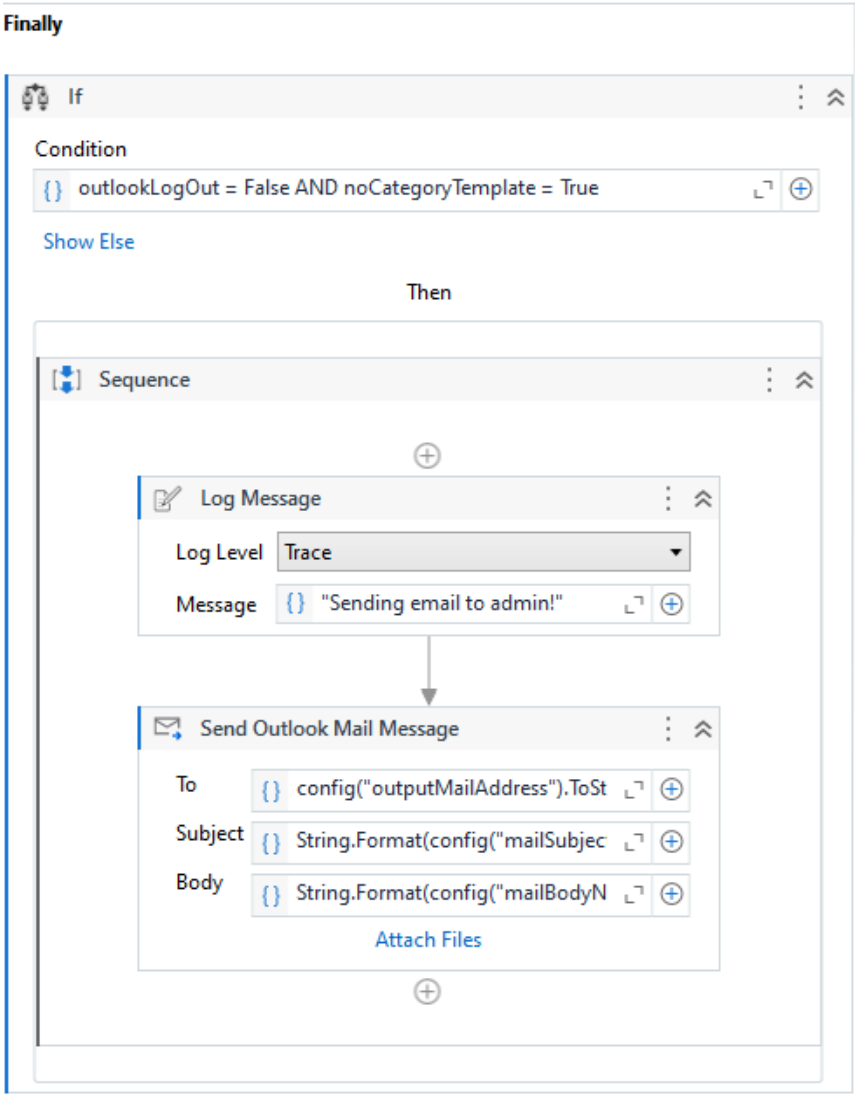
Posledním krokem celé této workflow je podmínka (umístěna mimo „For Each Row in Data Table“ cyklus). Aktivity uvnitř podmínky se vykonají, pokud se jedná o poslední e-mail v listu `listOfMails_Filtered`. Samotný kód začíná výpisem informace do logu o aktualizaci provedených změn. Důvodem je neschopnost dynamické změny v listu přečtených zpráv, robot tedy nastaví e-mailu kategorii, ale tuto změnu již nevidí (vidět lze pouze v Outlooku). Proto se zde opět využije workflow „Read_Mails“ pro přečtení všech zpráv viz 5.1.4. E-maily, které nemají kategorii,

jsou považovány za neznámé, jejich kategorizace již proběhla a jsou označeny jako „Unread/Nepřečtené“.



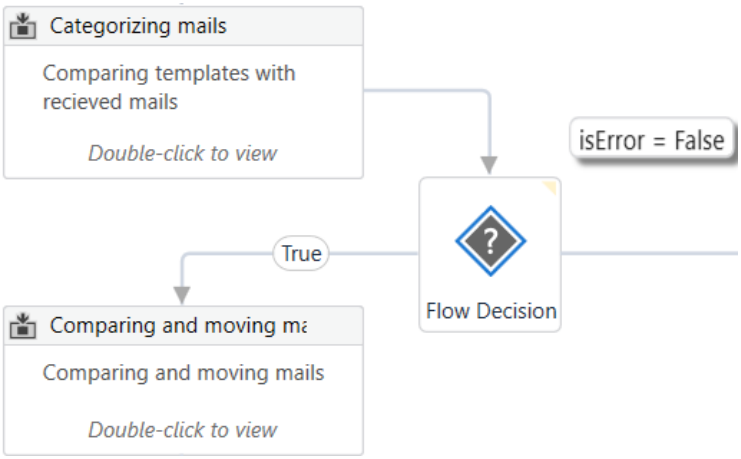
Obr. 5.15: Část workflow „mailCategorizer“

Tato část využívá i blok „Finally“ v aktivitě „Try Catch“. Ten se provede vždy po dokončení kategorizace. Kontroluje se hodnota proměnné `noCategoryTemplate`, v případě, kdy by byla nastavena jako `True`, robot zašle zprávu administrátorovi, že se v dané složce nachází šablona bez kategorie udávající četnost zprávy.



Obr. 5.16: Blok **Finally** po dokončení kategorizace

Dále je kontrolována proměnná `isError` (obrázek 5.17).



Obr. 5.17: Kontrolní podmínka hodnoty proměnně **isError**

5.1.6 Comparing and Moving mails (Porovnávání a přesouvání e-mailů)

Tento oddíl se věnuje porovnávání e-mailů stejného předmětu dle data příchodu. Zároveň ověří, jestli mezi jednotlivými zprávami nevzniklo zpoždění a případně kontaktuje administrátora. Poté jsou staré zprávy přesunuty do archivní složky, tak aby v hlavní složce zůstal pouze jeden e-mail.

Prvním krokem je přiřazení názvu složky proměnné `folderPath` z konfiguračního souboru. Poté je vyvolána workflow „mailOccurrence“, která kontroluje zpoždění mezi jednotlivými maily.

mailOccurrence

Tento soubor má 4 vstupní argumenty:

1. proměnnou se všemi šablony (`dtTemplates`),
2. slovník `mailDates`, definovaný v minulé pod-části 6,
3. list se všemi zprávami s jednou kategorií,
4. slovník `config`, který reprezentuje konfigurační soubor.

Vstupní aktivitou do tohoto workflow je For cyklus přes všechny e-maily, které při čtení schránky měli jednu kategorii. Podmínkou je zjištěno, jestli se ve slovníku `mailDates` nachází předmět zprávy ve formě klíče. Jestli se daný předmět ve slovníku nachází, tak je tato zpráva přidána do listu pod ním přístupným. Pokud se předmět ve slovníku nenachází, vytvoří se dočasný list `tempListMail` datového typu `mailMessages` a zpráva se do něj přidá. List je potom přidán jako hodnota ke klíči (předmět mailu).

Následující aktivitou je „For Each Row in Data Table“ přes řádky proměnné `dtTemplates`, tedy všechny šablony. První podmínkou v cyklu je zjištěno, jestli se

předmět šablony nachází uvnitř slovníku `mailDates`. V případě, že se zde nachází, jsou inicializované pomocné proměnné „tempList“ datového typu `list(Of String)` a „tempDict“ datového typu `Dictionary(of String, list(Of String))`. Pomocí dalšího „For Each“ cyklu se seřadí list se zprávami podle data příchodu a jeho nejstarší záznam se uloží do proměnné `actualDate`. Dále se pokračuje iterací přes seřazené zprávy podle data. V první iteraci se neprovádí nic, jelikož by se porovnávali dvě stejné zprávy. Od dalšího průchodu seznamu je podmínkou zjištěno jestli se v konfiguračním souboru nachází specifická časová tolerance (pro aktuální předmět), pokud tomu tak není, je využita výchozí hodnota. Ta se přičte datu příchodu předchozího mailu a porovná se s aktuální datem příchodu mailu v seznamu. Pro lepší pochopení viz zápis podmínky níže.

```
receivedDate <= actualDate.AddHours(tempTolerance)
```

Pokud vzniklo větší zpoždění, než je akceptováno, je o této situaci informován administrátor. Do proměnné `actualDate` se poté uloží zpráva novější a porovnávání se opakuje pro všechny zprávy listu.

Po dokončení kontroly dochvilnosti mailu je vyvolána workflow jménem „mailMover“ (mimo workflow „mailOccurrence“) a maily jsou přesouvány do archivních složek

mailMover

Tento soubor má 5 vstupních argumentů:

1. proměnnou se všemi šablony (`dtTemplates`),
2. slovník `mailDates`, definovaný v minulé pod-části 6,
3. archivní složku,
4. list se všemi zprávami s jednou kategorií.

Vstupní aktivitou je „For Each“ cyklus přes všechny šablony. První podmínkou v cyklu je zjištěno, jestli se předmět šablony nachází uvnitř slovníku `mailDates`.

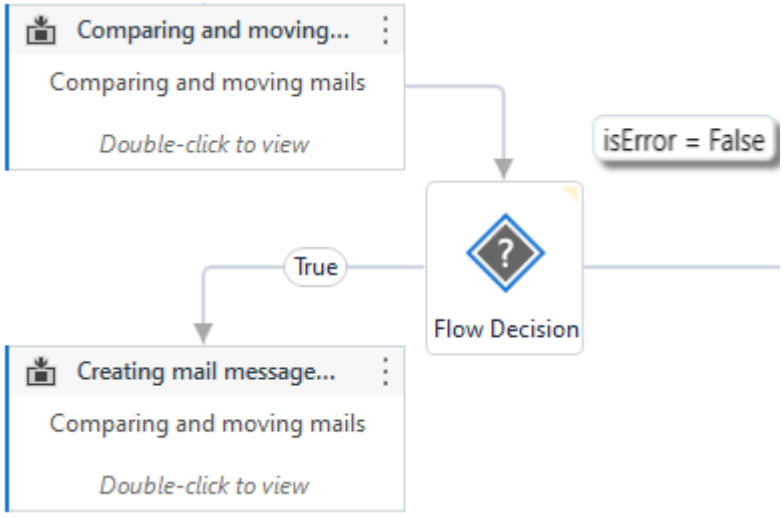
V případě, že se zde nachází, je zjištěna délka listu přístupná pomocí tohoto klíče. Pokud je délka rovna 1, tak se zpráva nepřesouvá (zůstane v dané složce), ale kontroluje se, jestli zpráva dorazila za posledních 24 hodin od doručení. Pokud tomu tak není (zpráva dorazila např. před 30 hodinami), je o této situaci informován administrátor.

Jestli je délka listu větší než 1, tak se do pomocného listu uloží list přístupný klíčem a do dočasné proměnné první hodnota z tohoto listu. Na to navazuje For cyklus přes všechny hodnoty. Podmínkou je zjištěno, jestli se index nerovná nule (takový index je přeskočen z důvodu porovnávání stejné zprávy). Jakmile je hodnota větší než 0, je pomocí aktivity „Compare date“ zjištěno, která ze zpráv je ve schránce

delší dobu (starší) a ta je přesunuta do archivu. Do dočasné proměnné je nahrána zpráva novější a porovnávání se opakuje do doby, kdy ve schránce zůstane jeden e-mail v rámci jednoho předmětu.

Po přesunutí mailů je i zde zkontrolováno jestli zpráva dorazila za posledních 24 hodin od doručení. Pokud tomu tak není, je o této situaci informován administrátor.

I po této pod-části je kontrolována proměnná `isError` pro případnou chybu při přesouvání zpráv.



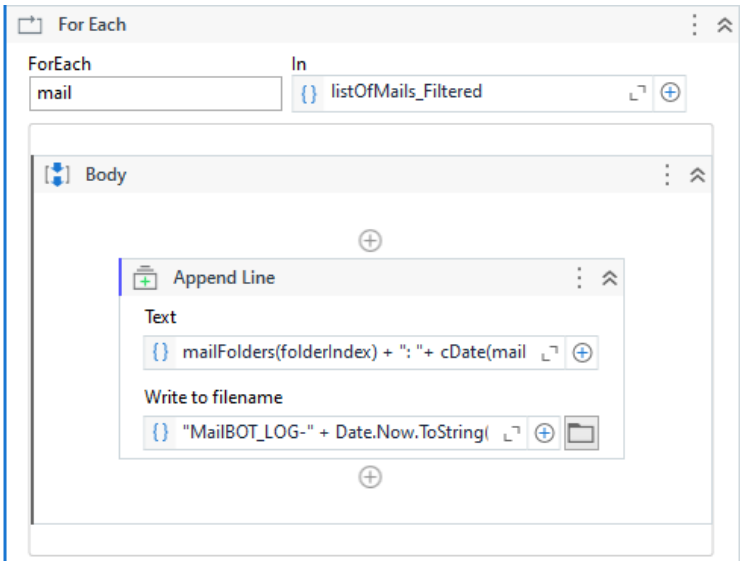
Obr. 5.18: Kontrola hodnoty proměnné **isError** pro případ chyby

5.1.7 Creating Mail Message and Log (Vytváření zprávy a logu)

Tato část se zabývá vytvořením zprávy pro administrátora a logu. Vstupními aktivitami se inicializují proměnné a podmínkou se zkontroluje, jestli byl přečten nějaký e-mail. Pokud nebyl, tak se do proměnné `mailCounter` nastaví 0. Pokud nějaký mail přečten byl, tak se s proměnnou nic dále neděje. Dále se využije zpráva z konfiguračního souboru, do které se na místa {0}{1}{2} dosadí skutečné hodnoty viz 5.1.7.

```
mailBody= at folder: {0} was {1} regular mails and {2} unknown mails,
mailBody= at folder: Inbox was 15 regular mails and 5 unknown mails
```

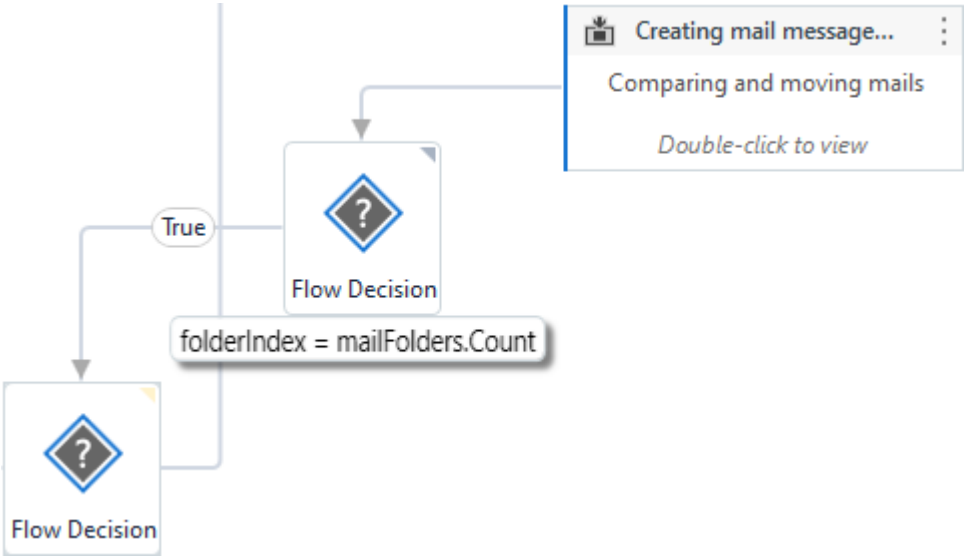
Zpráva se uloží do pole `mailMessages`. Dále se pomocí For cyklu vypíše do log souboru všechny nově příchozí maily 5.19.



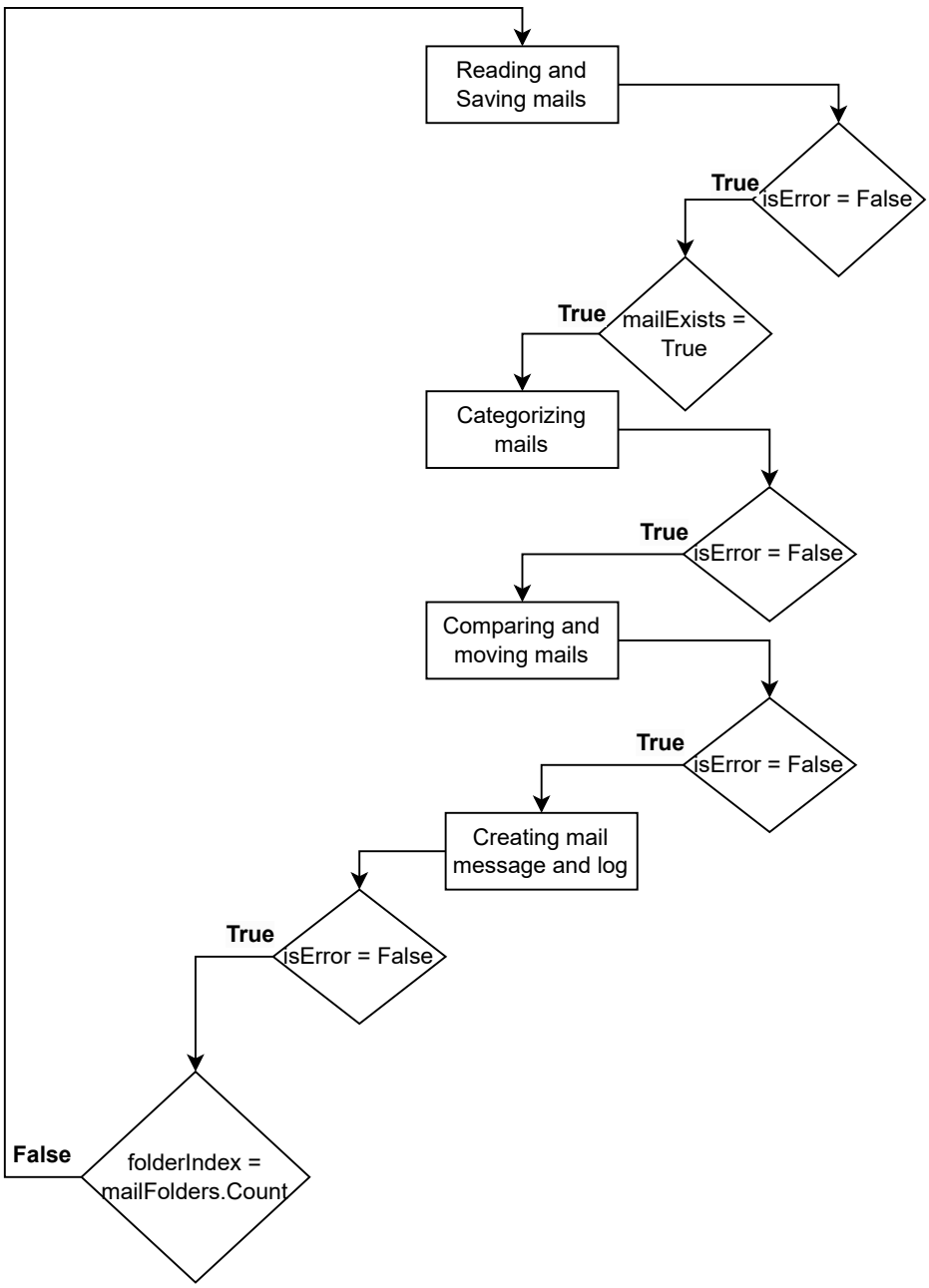
Obr. 5.19: Vytváření logu

Posledním krokem je zvýšení hodnoty proměnné `folderIndex`, který reprezentuje index v listu 5.1.2, jenž v sobě uchovává názvy složek, které robot zpracovává.

Po dokončení tohoto bloku kódu následuje řídící podmínka (tak jak jde vidět na obrázku 5.20), která kontroluje jestli se délka listu `mailFolders` rovná hodnotě indexu `folderIndex`. Pokud tomu tak není, tak se robot přesune na část „Reading and Saving mails“ pro další hodnotu v listu `mailFolders`, tedy další složku, kterou bude robot kategorizovat. Pro lepší představu slouží diagram 5.21. Po dokončení kategorizace pro všechny složky následuje část pro zasílání notifikace administrátorovi.



Obr. 5.20: Podmínka pro kontrolu další složky pro kategorizaci



Obr. 5.21: Diagram robota - části, které se opakují pro každou složku

5.1.8 Email Notification (Notifikace e-mailem)

Posledním krokem robota je notifikace administrátora o výsledku průběhu procesu. V této části je podmínkami zjištěno, zdali robot úspěšně dokončil proces, či vznikla chyba v jeho vykonání.

První podmínkou je kontrola hodnoty proměnné `isError`, pokud je její hodnota `True`, tak je další podmínkou zjištěno, jestli je `Outlook` přihlášen. Pokud ano, jsou proměnným z konfiguračního souboru přiřazeny hodnoty (odesilatel, příjemce, tělo e-mailu). Bot poté zašle e-mail administrátorovi prostřednictvím aktivity „Send Outlook Mail Message“. Výběr této aktivity je opět z důvodu nepotřebné autentizace, využívá již přihlášený `Outlook` 5.22. V případě odhlášeného Outlooku se pouze vypíše informace do logu a robot se ukončí.

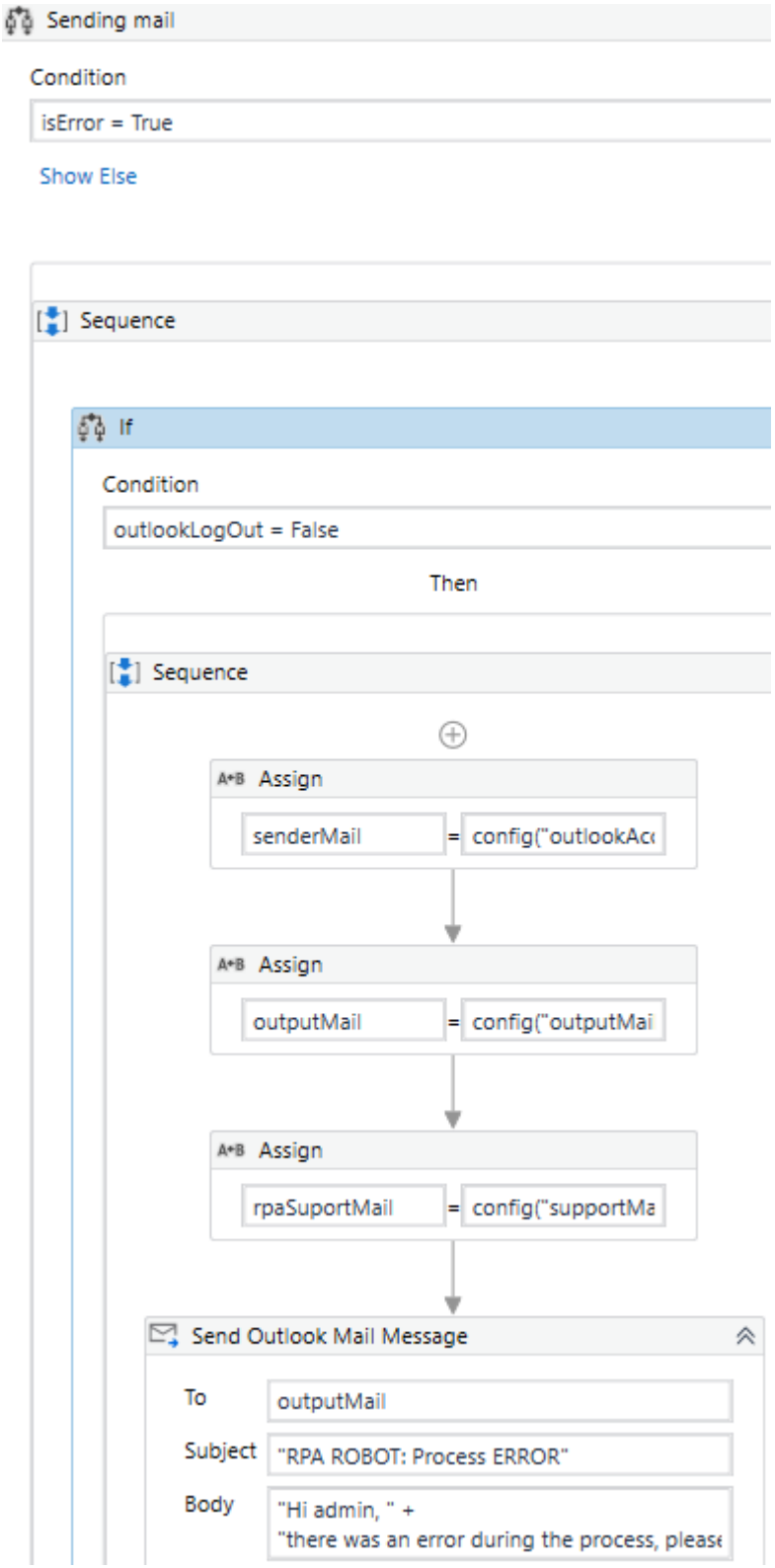
Další podmínka (obrázek 5.23) je pro ověření, jestli je `Outlook` přihlášen, k tomu slouží proměnná `outlookLogOut`. V případě odhlášeného Outlooku robot vypíše informace do logu a je ukončen.

Třetí podmínka je pro zjištění, zdali byla přečtena nějaká zpráva. Pokud nebyla, je o této skutečnosti informován administrátor prostřednictvím e-mailu viz obrázek 5.24.

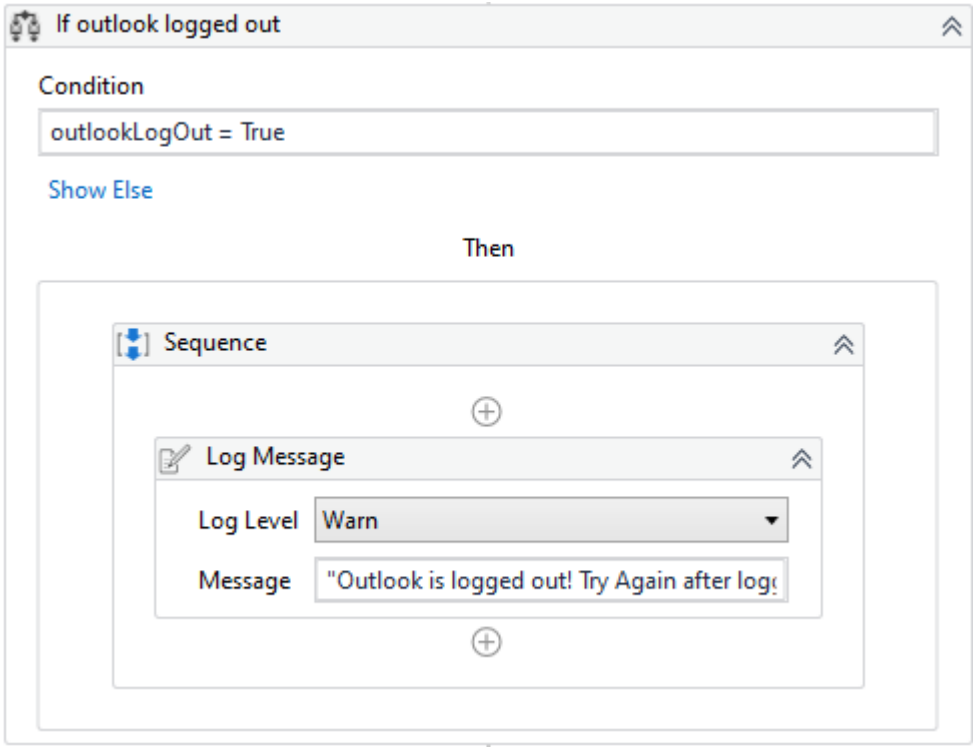
Poslední podmínka je pro případ úspěšného dokončení procesu. Pomocí For cyklu přes pole `mailMessage`, obsahující informace ze všech složek, je poskládaná výsledná zpráva. Ze zaslané zprávy je na první pohled jasné, kolik přišlo nových zpráv a kolik z nich je neznámých.

Před samotným ukončením robota je využito workflow „KillAllProcess“ pro vypnutí Outlooku.

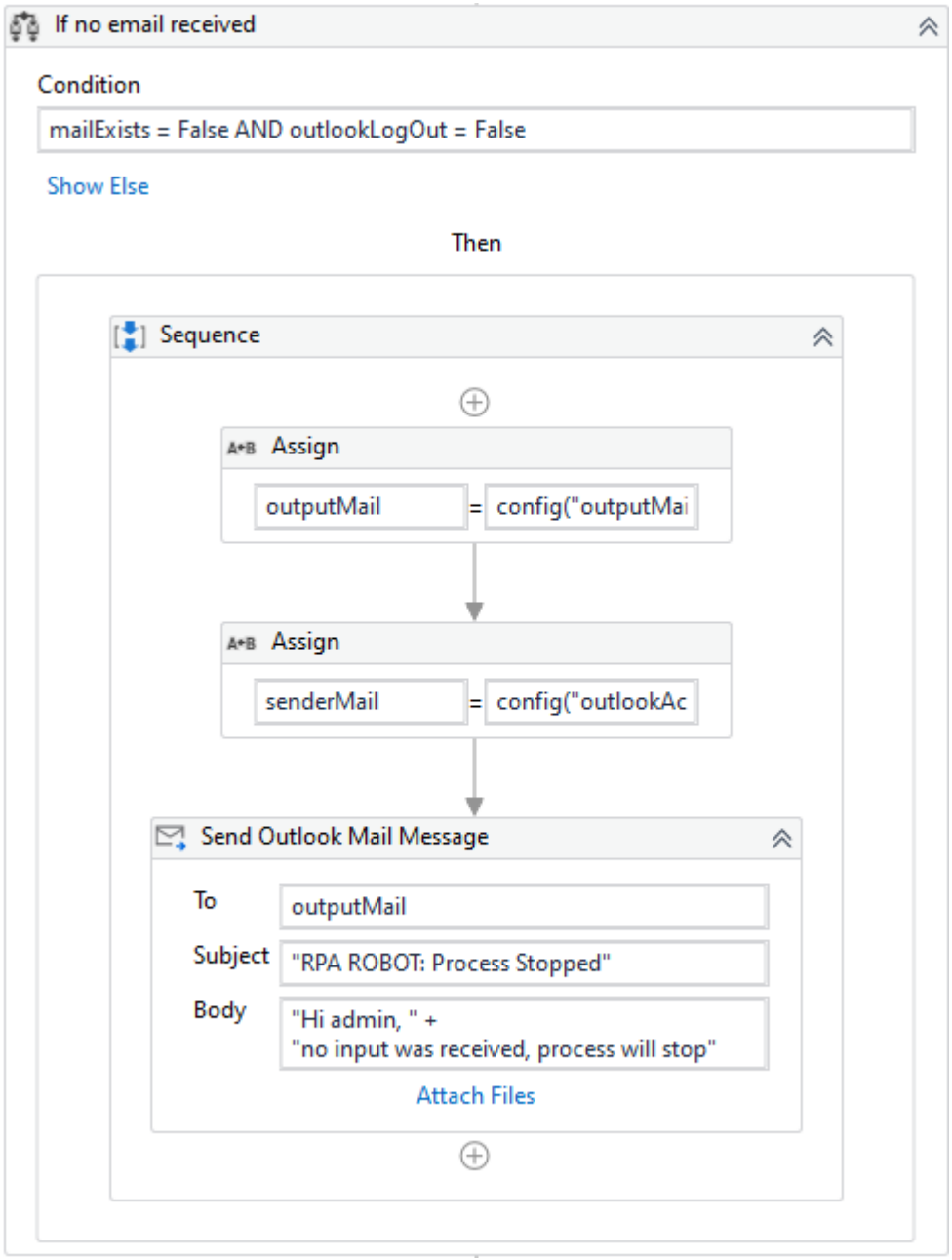
V této části je blok **Catch** obohacen o odeslání e-mailu s informacemi o výjimce, tak jak jde vidět na obrázku 5.25.



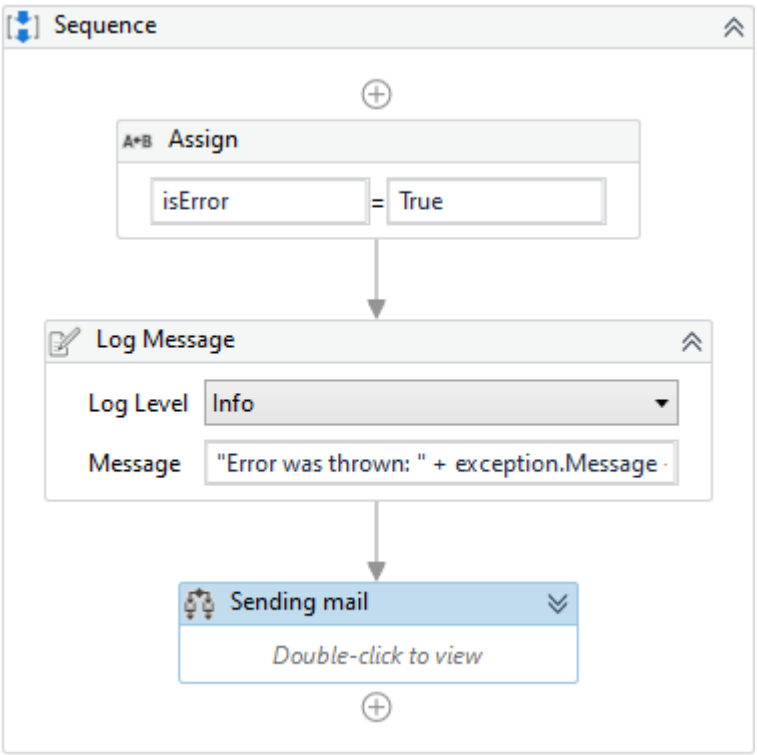
Obr. 5.22: Podmínka pro případ chyby s přihlášeným Outlookem



Obr. 5.23: Podmínka pro případ odhlášeného Outlooku



Obr. 5.24: Případ kdy nejsou ve schránce žádné e-maily



Obr. 5.25: Upravený blok **Catch**

Závěr

Tato práce měla za úkol uvést čtenáře do problematiky automatizací za pomocí robotických procesů (RPA). Praktickou částí bylo vytvoření robota pro automatizovanou kategorizaci schránky.

Vymezené cíle byly splněny, čtenář dokáže popsat referenční model ISO/OSI a jeho návaznost na TCP/IP. Dále fungování počítačových sítí a základních poštovních protokolů. Taktéž rozumí pojmům z oblasti automatizací, zejména pak softwarových robotů. Pozná základní vývojové prostředí pro vývoj RPA a jejich rozdíly.

Samotná implementace byla taktéž splněna i přes různé překážky. Například výběr aktivit pro přístup k poštovní schránce, architektura robota či případné zvolení datových typů pro ukládání dat.

Během vývoje se vyskytly určité problémy, například neschopnost robota rozpoznat `Outlook` ve více jazycích, přesouvání již kategorizovaných e-mailů nebo nefunkční kontrola dochvilnosti zpráv. Všechny tyto problémy byly v průběhu vývoje vyřešeny a robot funguje bez potíží.

Literatura

[1] PAMELA, F. : *Počítačové sítě* [online]. URL: <https://www.khanacademy.org/computing/computers-and-internet/xcae6f4a7ff015e7d:the-internet/xcae6f4a7ff015e7d:connecting-networks/a/computer-networks-overview> [Navštíveno 15. 11. 2022]

[2] MACHALÍK, F., VESELÝ, P., MACHALÍK, S., SADLOŇ, L.: *Úvod do informačních technologií - Počítačové sítě* [online]. 2008. URL: <https://ct.upce.cz/machalik/puitk-stare/text.htm> [Navštíveno 15. 11. 2022]

[3] VOJTĚŠEK, J.,: *Internet a jeho služby – Teferenční model ISO/OSI.* Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky. [online] 2012. URL: <http://ijs2.8u.cz/index.php?option=com_content&view=article&id=13&Itemid=119> [Navštíveno 15. 11. 2022]

[4] ČÍKA, P.: *Multimediální služby.* Brno: Vysoké Učení Technické, Fakulta elektrotechniky a komunikačních technologií, 2012. 8-9 s. ISBN 978-80-214-4443-0. [Navštíveno 15. 11. 2022]

[5] JEŘÁBEK, J.: *Komunikační technologie.* Brno: Vysoké Učení Technické, Fakulta elektrotechniky a komunikačních technologií, 2013. Naposledy upraveno 20. 08. 2020. ISBN 978-80-214-4713-4. [Navštíveno 15. 11. 2022]

[6] VOJTĚŠEK, J.,: *Internet a jeho služby – Srovnání RM ISO/OSi a TCP/IP.* Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky. [online] 2012. URL: <http://ijs2.8u.cz/index.php?option=com_content&view=article&id=15&Itemid=121> [Navštíveno 16. 11. 2022]

[7] *Co je to IMAP a co POP?* [online]. URL: <https://support.microsoft.com/cs-cz/office/co-je-to-imap-a-co-pop-ca2c5799-49f9-4079-aefe-ddca85d5b1c9> [Navštíveno 19. 11. 2022]

[8] *Jaký je rozdíl mezi protokoly POP a IMAP?* [online]. URL: <https://support.microsoft.com/cs-cz/office/jak%C3%BD-je-rozd%C3%ADl-mezi-protokoly-pop-a-imap-85c0e47f-931d-4035-b409-af3318b194a8> [Navštíveno 19. 11. 2022]

[9] FORGÁČ, J.: *Jaký protokol použít IMAP, nebo POP3* [online]. 18.8.2017. URL: <https://www.artweby.cz/blog/jaky-protokol-pouzit-imap-nebo-pop3> [Navštíveno 19. 11. 2022]

[10] VOJTĚŠEK, J.,: *Internet a jeho služby – Aktivní síťové prvky.* Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky. [online] 2012. URL: <https://ijs2.8u.cz/index.php?option=com_content&view=article&id=18&Itemid=123> [Navštíveno 19. 11. 2022]

[11] VOJTĚŠEK, J.,: *Internet a jeho služby – Pasivní síťové prvky.* Univerzita Tomáše Bati ve Zlíně, Fakulta aplikované informatiky. [online] 2012. URL: <https://ijs2.8u.cz/index.php?option=com_content&view=article&id=19&Itemid=124> [Navštíveno 19. 11. 2022]

[12] SCOTT, B.: *What is a Process?* [online] URL: <https://www.processmodel.com/blog/what-is-a-process/> [Navštíveno 20. 11. 2022]

[13] SAM, D.: *Robotics Technology* [online] Naposledy upraveno 18. 08. 2022. URL: <https://builtin.com/robotics>[Navštíveno 20. 11. 2022]

[14] *3 Nejčastější možnosti automatizace výroby* [online]. URL: <https://factoryautomation.cz/3-nejcastejsi-moznosti-automatizace-vyroby/> [Navštíveno 19. 11. 2022]

[15] *Co je RPA?* [online]. URL: <https://powerautomate.microsoft.com/cs-cz/what-is-rpa/> [Navštíveno 19. 11. 2022]

[16] OBST, M.: *Použití RPA v telekomunikační společnosti.* Bakalářská práce. Univerzita Hradec Králové, Fakulta informatiky a managementu. [online]. Hradec Králové, 2020. URL: <https://theses.cz/id/93nrwn/> [Navštíveno 20. 11. 2022]

[17] RAVINDRA, S.: *Blue Prism Vs UiPath* [online]. URL: <https://mindmajix.com/blue-prism-vs-uipath> [Navštíveno 22. 11. 2022].

[18] RAVINDRA, S.: *What is Network Monitoring?* [online]. URL: <https://www.vmware.com/topics/glossary/content/network-monitoring.html> [Navštíveno 15. 03. 2023].

[19] BOUŠKA, P.: *SNMP - Simple Network Management Protocol* [online]. 20.12.2016. URL: <https://www.samuraj-cz.com/clanek/snmp-simple\-network-management-protocol/> [Navštíveno 15. 03. 2023]

[20] IVORA, A.: *SNMP a monitoring sítě* [online] 2020. URL: <https://www.fi.muni.cz/~kas/pv090/referaty/2020-podzim/snmp.html> [Navštíveno 15. 03. 2023]

Seznam symbolů a zkratek

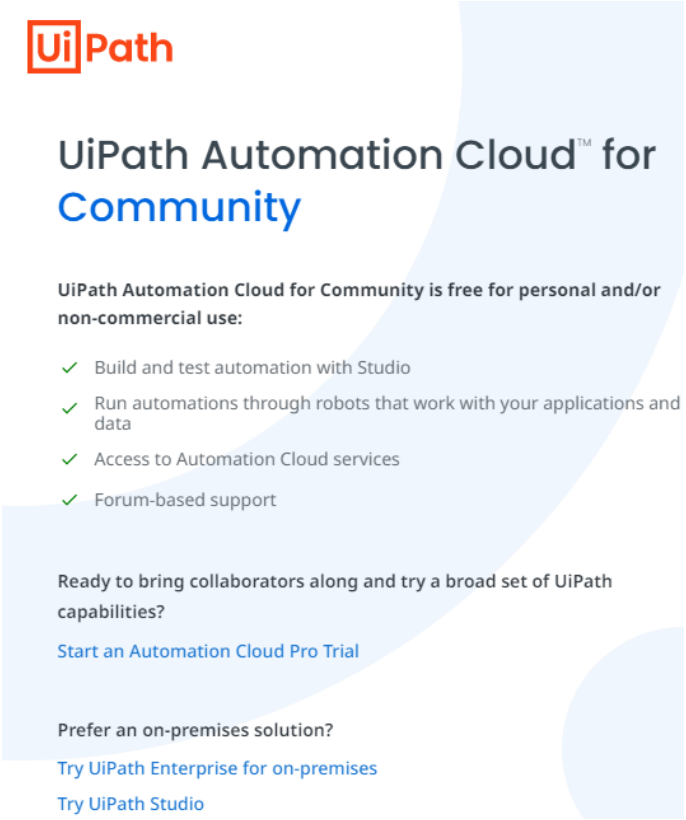
TCP/IP Transmission Control Protocol/Internet Protocol
ISO/OSI International Standards Organisation / Open System Intercon-nection
TCP Transmission Cotrol Protocol
UDP User Datagram Protocol
SMTP Simple Mail Transfer Protocol
HTTP Hypertext Transfer Protocol
FTP File Transfer Protocol
POP3 Post Office Protocol 3
IMAP4 Internet Message Access Protocol 4
WAN World Area Network
LAN Local Arean Network
MAC Media Access Control
RPA Robotic Process Automation
SNMP Simple Network Management Protocol
ICMP Internet Control Message Protocol

Seznam příloh

A Manuál k nasazení robota

A.1 Nastavení UiPath Assistant

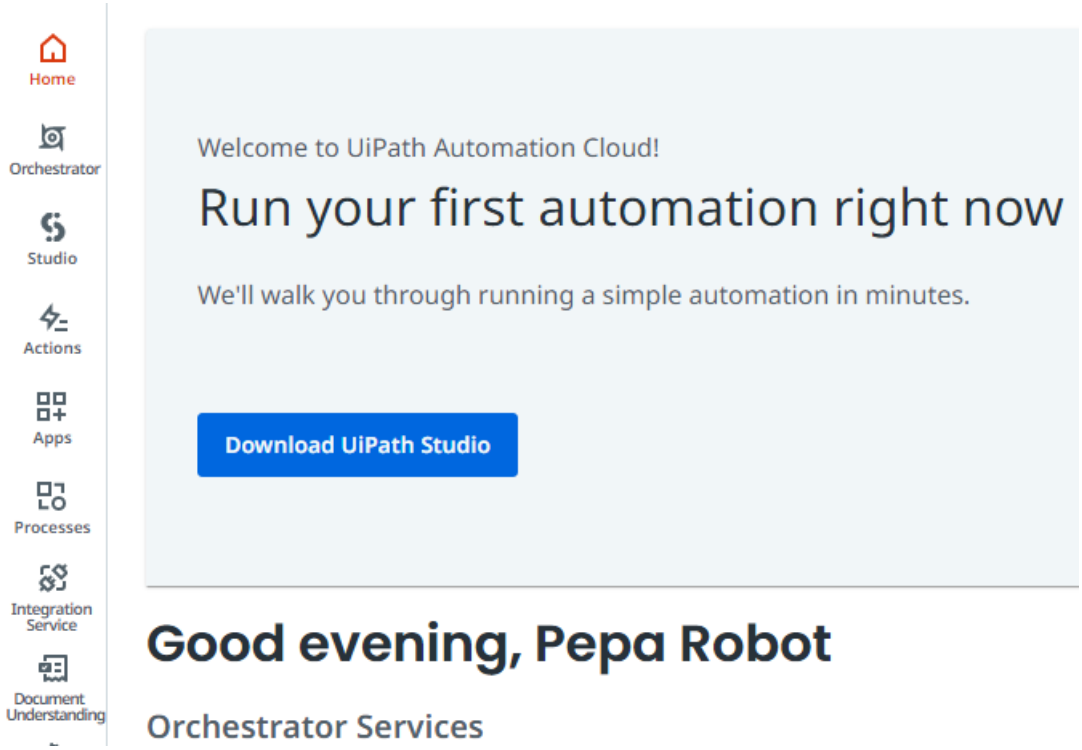
Pro správné nastavení stanice, na které bude robot pracovat, je nutná registrace na stránkách UiPath. Pro bezplatný účet je nutné vybrat „UiPath Automation Cloud for Community“, tak jako jde vidět na obrázku.



Obr. A.1: Výběr typu účtu - komunitní edice

1. Při prvním přihlášení stránka vyzve k vytvoření tzv. `Automation Cloud`, prosím potvrďte, je to totiž nezbytné pro přístup do Orchestratoru.
2. Nyní stáhněte aplikaci `UiPath Studio`, kliknutím na modré tlačítko **Download UiPath Studio**.
3. Otevřete instalační soubor a klikněte na možnost „Custom (recommended for Enterprise/Advanced Users)“
4. Na další straně klikněte na „Install for me only“ a pokračujte na výběr balíku. Zde vyberte možnost „Attended Robot“, pokud chcete robota spouštět bez možnosti pohledu do kódu. [1]

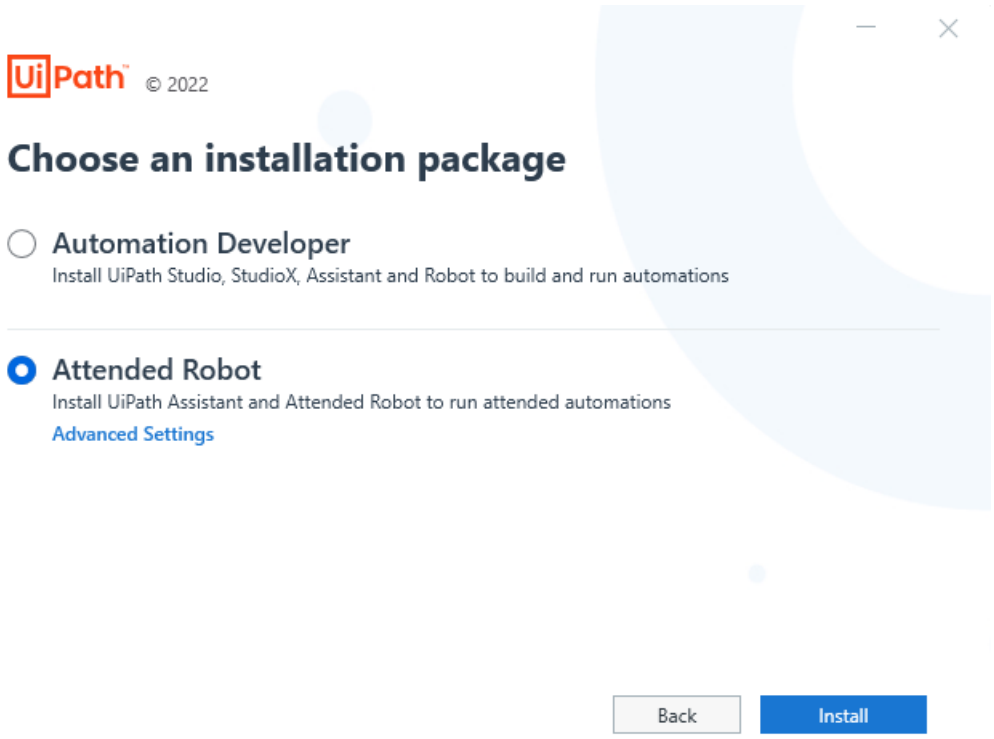
[1] Dostupné na https://www.uipath.com/



Obr. A.2: Stránka po správném přihlášení a vytvoření **Automation Cloud**

5. Po instalaci `UiPath Assistant` je nutné tuto aplikaci spojit s webovým cloudem `Orchestrator`. Přihlaste se pomocí modrého tlačítka **Sign in**. Budete přesměrováni na stránky UiPath, zde povolte „Otevření aplikace UiPath“. [2]
6. Pokud bylo vše správně provedeno, tečka vedle jména účtu je zelená.

[2] UiPath – www.uipath.com



Obr. A.3: Výběr instalace - **Attended Robot**

Obr. A.4: Správné připojení Orchestratoru

A.2 Úprava konfiguračního souboru

Důležitým krokem, bez kterého by vám robot nefungoval, je úprava údajů, které bude robot využívat.

1. Rozbalte balíček s kódem. (soubor .nupkg většinou rozbalit nelze, stačí ho přejmenovat na .zip a pokračovat klasickým způsobem)
2. Uložte soubor „Config“.
   (`<Nazev_balicku>\lib\net45\Config.xlsx` - doplňte o svůj adresář)
3. Soubor otevřete a upravte údaje v zeleně zbarvených buňkách podle vlastních údajů (**hodnoty** v tabulkách **nemůžou** být ve formě hypertextového odkazu!).

A.3 Přidání aktiv

Po úspěšném doplnění vlastních údajů do konfiguračního souboru je nutné uložit jeho cestu do webového cloudu `Orchestrator`.

1. Ve webovém cloudu `Orchestrator` se přesuňte do Vámi vybrané složky, např. „My Workspace“.
2. V sekci „Assets“ modrým tlačítkem **Add Asset -> Create a new asset** přidejte dvě aktiva, které robot využívá.
3. Prvním aktivem je cesta ke konfiguračnímu souboru, který se musí jmenovat jako **Outlook Category Robot Config**. Do kolonky „Value“ zadejte Vaši cestu k tomuto souboru.
4. Vytvoření potvrďte tlačítkem **Create**.
5. Jako druhé aktivum je prostředí, v kterém robot pracuje, typicky jsou 3 - vývoj, test a produkce. Jméno aktiva je **Environment**, hodnotu nastavte podle prostředí, v kterém se bude robot spouštět, např. Produkce.

A.4 Nastavení Orchestrator

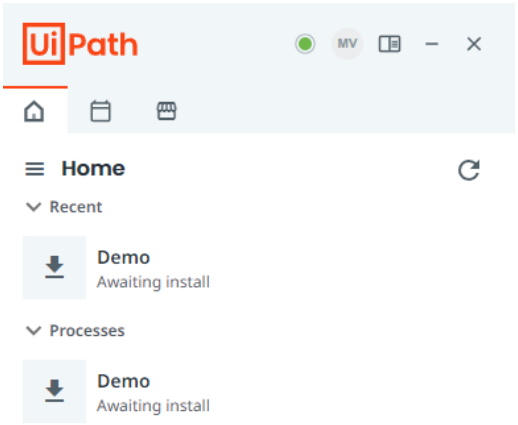
Robota lze spustit dvěma způsoby:

A.4.1 Attended

Bot je obsluhován člověkem. Proces probíhá na hlavní stanici uživatele, kde je nainstalovaný a připojený `UiPath Assistant`. Tento způsob je jednodušší.

Přidání balíku a vytvoření procesu

1. Ve webovém cloudu `Orchestrator` se přesuňte do složky „My Workspace“.
2. V sekci „Automations“, v podkategorii „My Packages“ modrým tlačítkem „Upload“ přidejte balíček s kódem.
3. Přesuňte se do podkategorie „Processes“.
4. Klikněte na modré tlačítko **Add Process** a vyberte vámi přidaný balíček,
5. Dále pokračujte na další stranu, kde doplňte všechny potřebné aktiva, které balíček využívá,
6. Na poslední straně jsou doplňující nastavení, jako např. jméno procesu, popisek či priorita spouštění.
7. Nastavení potvrďte tlačítkem **Create**.
8. Nyní je vše nastavené a můžete robota spustit.



Obr. A.5: **UiPath Assitant** s procesem „Demo“

Spuštění robota

1. Otevřete program `UiPath Assistant`.
2. Na domovské stránce, byste měli vidět Vámi přidaný proces, např. „Demo“.
3. Proces spustíte pomocí tlačítka **Run**.

A.4.2 Unattended

Bot nevyžaduje lidskou interakci. Je nutné vytvoření účtu robota a přidání přihlašovacích údajů k účtu Windows v nastavení Orchestratoru. Stanice, na které robot vykonává proces, musí být připojená pomocí `UiPath Assistant`, nejčastěji na virtuálním stroji.

Přidání účtu robota

1. Přesuňte se do záložky `Tentant`.
2. V podkategorii „Manage Access“ klikněte na modré tlačítko **Manage Accounts & Groups**.
3. Na „Administration“ portálu v kategorii „Robot accounts“ vytvořte účet pro robota.

Licence účtu

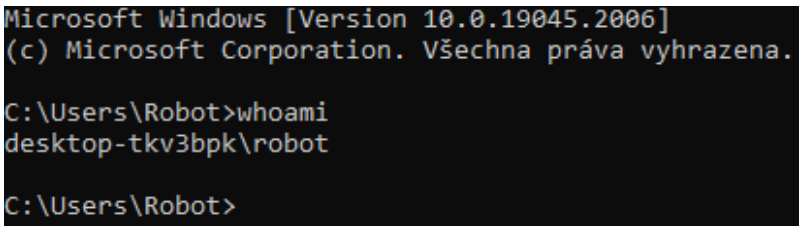
Práva na spouštění robotů jsou řízeny licencemi, proto musíte příslušnou licenci k vašemu účtu přiřadit.

1. V podkategorii „Manage Acess“ klikněte na tři tečky u svého účtu a dále na **Edit**.
2. V sekci „Unattended setup“ povolte spouštění tohoto typu robotů.

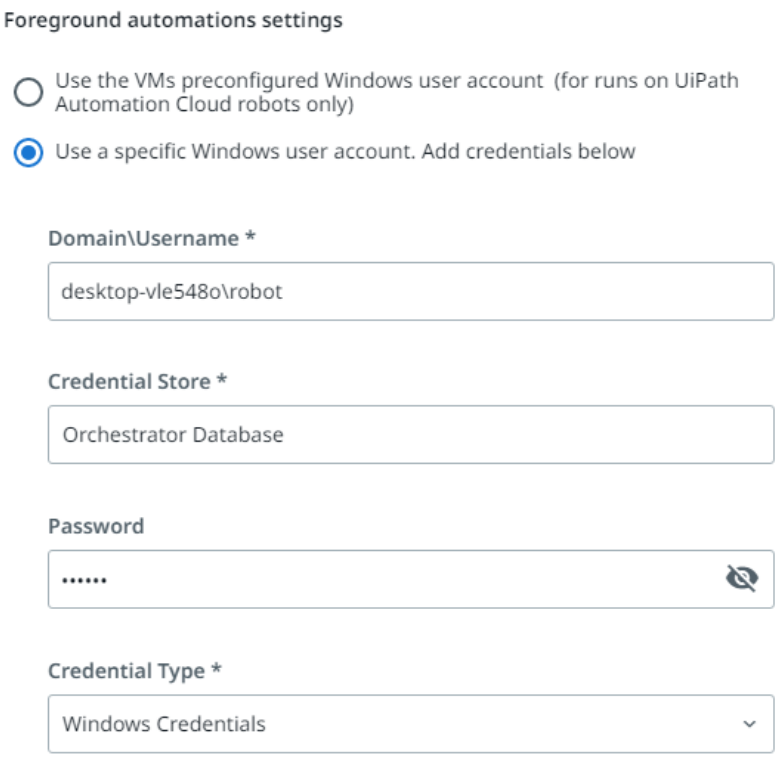
3. Změny uložte tlačítkem **Update**.

Přidání účtu robota do Orchestratoru

1. Ve stejné podkategorii jako v předešlém kroku přidejte nově vytvořený účet robota.
2. **Assign roles** -> **Robot account**, do vyhledávacího okna napište jméno vašeho robota a přiřaďte mu roli „Robot“.
3. Přejděte na další stranu a v sekci „Foreground automations“ zaškrtněte možnost „Use a specific Windows user account. Add credentials below“ a přidejte přihlašovací údaje, pod kterými se robot přihlásí. („Domain Username“ zjistíte příkazem `whoami` v příkazové řádce)



Obr. A.6: Příkazová řádka - domain username

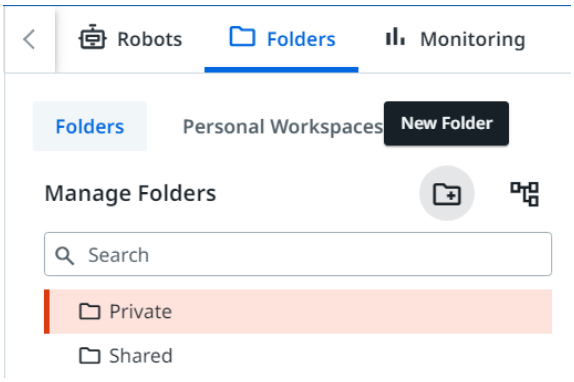


Obr. A.7: Nastavení popřední automatizace - přidání Windows účtu

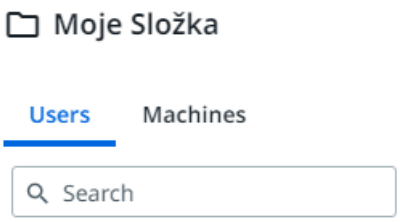
4. Přidání dokončíte tlačítkem **Skip and assign**.

Vytvoření složky

1. V Orchestratoru se přenuťe do záložky `Tentant`, do podkategorie **Folders**.
2. Vytvořte složku s libovolným názvem, např. „Moje složka“.
3. Klikněte na vytvořenou složku v seznamu „Manage Folders“.
4. K složce přiřaďte účet robota v záložce „Users“ pomocí tlačítka **Assign Account/Group**.



Obr. A.8: Vytvoření složky



Obr. A.9: Záložky Users a Machines

5. Vyplňte jméno robota a přiřaďte roli.
6. Stejný postup zopakujte pro přidání stanice v záložce „Machines“.
7. Pokud vám po přidání stanice u jména svítí vykřičník, klikněte u ní na **tři tečky -> Edit Machine** -> v sekci „Runtime details“ zvolte **Production (Unattended) na 1**
8. Tlačítkem **Update** potvrďte.

Přidání balíku a vytvoření procesu

1. Ve webovém cloudu `Orchestrator` se přesuňte do záložky `Tenant`.
2. V sekci „Packages“ modrým tlačítkem „Upload“ přidejte balíček s kódem.
3. Přesuňte se do přidané složky do podkategorie „Processes“.
4. Klikněte na modré tlačítko **Add Process** a vyberte vámi přidaný balíček,
5. Dále pokračujte na další stranu, kde doplňte všechny potřebné aktiva, které balíček využívá viz A.3 (případně můžete využít i již přidané aktiva v jiných složkách tlačítkem **Import asset**),
6. Na poslední straně jsou doplňující nastavení, jako např. jméno procesu, popisek či priorita spouštění.
7. Nastavení potvrďte tlačítkem **Create**.
8. Nyní je vše nastavené a můžete robota spustit.

Spouštění robota

1. V Orchestratoru se přesuňte do vytvořené složky, podkategorie „Automations“, sekce **Jobs**.
2. Pomocí modrého tlačítka **Start** vytvořte úkol.
3. Z nabídky procesů jeden vyberte, nastavte prioritu a prostředí ve kterém se robot spouští, např. Vývoj.
4. Jako účet zvolte robota a potvrďte **Start** tlačítkem.

B Obsah elektronické přílohy

- `BP_OutlookBOT` ..... adresář s robotem
  - `_rels`
  - `lib`
  - `net45` ..... adresář s použitými workflow's
    - `.objects`
    - `.settings`
    - `.project`
    - `Config.xlsx` ..... konfigurační soubor robota
    - `Main.xaml` ..... hlavní (spustitelná) workflow
    - `InitAllSettings.xaml` ..... workflow pro načtení konfiguračního souboru
    - `KillAllProcesses.xaml` ..... workflow pro zavření programů
    - `outlookLogin.xaml` ..... worklow pro přihlášení Outlooku
    - `Read_Mail.xaml` ..... wokflow pro čtení e-mailů
    - `mailCategorizer.xaml` ..... workflow pro kategorizování e-mailů
    - `mailOccurrence.xaml` ..... workflow pro kontrolu dochvilnosti zpráv
    - `mailMover.xaml` ..... workflow k přesouvání zpráv
    - `project.json` ..... soubor obsahující informace o kódu
  - `package`
  - `content_types.xml`
  - `BP_OutlookBOT.nuspec`