Mittuniversitetet
MID SWEDEN UNIVERSITY

Processautomation och datautvinning

Övervakning med robotic process automation (RPA) och dess möjligheter

Ina Nilsson

Datateknik examensarbete, kandidatexamen
Huvudområde: Datateknik
Poäng: 15
Termin/År: VT23
Handledare: Karl Pettersson
Examinator: Patrik Österberg
Kurskod: DT099G
Program: Datateknik 180 hp

At Mid Sweden University, it is possible to publish the thesis in full text in DiVA (see appendix for publishing conditions). The publication is open access, which means that the work will be freely available to read and download online. This increases the dissemination and visibility of the degree project.

Open access is becoming the norm for disseminating scientific information online. Mid Sweden University recommends both researchers and students to publish their work open access.

I/we allow publishing in full text (free available online, open access):

☒ Yes, I/we agree to the terms of publication.

☐ No, I/we do not accept that my independent work is published in the public interface in DiVA (only archiving in DiVA).

Sundsvall, 2023-06-11 .....
Location and date

Datateknik, 180 hp .....
Programme/Course

Ina Nilsson .....
Name (all authors names)

1999-01-16 .....
Year of birth (all authors year of birth)

Sammanfattning

Robotic process automation (RPA) är en växande trend inom alla möjliga arbetsområden. En RPA-mjukvara har kapaciteten att utföra enformiga digitala uppgifter på ett nästintill mänskligt sätt tack vare dess förmåga att jobba mot alla typer av system och gränssnitt. Potentialen är enorm, däribland för övervakning och datautvinning. Syftet med detta projekt var att utforska RPA:ns möjligheter gällande övervakning, samt dess framtida potential. För det byggdes en så kallad RPA-robot i utvecklingsmiljön UiPath Studio för att automatisera processen att hämta väderinformation från tre olika väderwebbsidor, och testa robotens styrkor och svagheter. Resultatet blev en färdig robot som uppnådde kraven som ställts på den, samt visade det på en bra prestanda vid typiska förutsättningar där en körning genomsnittligen låg på under en minut. Det visade dock också att RPA har tydliga brister, som vid förändringar av struktur och diverse störningar, och det visade att implementation spelar väldigt stor roll i hur roboten hanterar diverse situationer. Det som härledes från resultatet var att RPA är en något stel men samtidigt otroligt användbar teknologi som är bra för det den är ämnad för. Den har stor potential, särskilt i kombination med andra teknologier som artificiell intelligens, som kan täcka RPA:ns begränsningar, och ämnar sig utmärkt till alla typer av processer.

Nyckelord: Robotic process automation (RPA), datautvinning, övervakning, processautomation, UiPath.

Abstract

Robotic process automation (RPA) is a growing trend withing all fields of work. An RPA-software has the capacity to perform monotonous digital tasks in a nearly human manner thanks to its ability to work across all systems and interfaces. The potential is huge, for surveillance and data mining among all other areas. The purpose of this project was to explore the possibilities of RPA regarding surveillance, as well as its future potential. For that, a so-called RPA-robot was built in the tool UiPath Studio to automate the process of fetching weather information from three different weather websites, and to test the strengths and weaknesses of the robot. The result was a finished robot which reached the established requirements and showed good performance under typical circumstances, where one run of the robot on average took under one minute. However, it also showed that RPA has clear flaws, such as when structure changes and when affected by various disturbances. What can be deduced from the result was that RPA is a somewhat stiff but also incredibly useful technology which is good for what it is meant for. It has great potential, especially in combination with other technologies like artificial intelligence, which can fill the gaps of RPA's limitations, and it works well for all types of processes, among them data mining and surveillance.

Keywords: Robotic process automation (RPA), data mining, surveillance, process automation, UiPath.

Innehållsförteckning

Terminologi

Förkortningar

RPA	Robotic process automation

1 Introduktion

Syftet med detta projekt är att utforska möjligheterna gällande övervakning med hjälp av teknologin robotic process automation (RPA). Den generella idéen grundar sig i RPA:s nutida och framtida relevans och var ett förslag från uppdragsgivaren Knowit, ett konsultbolag som riktar sig mot företag och organisationer med sina digitala lösningar.

Under detta kapitel kommer bakgrunden för teknologin och uppgiften att förklaras, samt mer ingående vad exakt som ska utföras.

1.1 Bakgrund och motivation

Automation av arbetsflöden har sedan 1990-talet varit en växande trend. Det grundläggande konceptet som RPA bygger på, någonting som kallas screen scraping och är en form av data scraping, har funnits sedan 1960-talet. RPA som specifik teknik är dock relativt nytt; begreppet robotic process automation uppstod först tidigt 2000-tal och konceptet är fortsatt under utveckling. [1]

En RPA-mjukvara är en så kallad robot som utför digitala uppgifter tidigare hanterade av människor, såsom inläsning av dokument, hämtning av data och startande av processer. Fördelen med RPA som särskiljer den från andra automationstekniker ligger i att denna robot agerar som en person och därmed kan genomföra samma digitala uppgifter som en person kan, oavsett gränssnitt och system. Samma robot kan också förflytta sig mellan olika gränssnitt inom samma arbetsflöde. [1] Detta innebär otroliga möjligheter för arbetsplatsen, där människor, i stället för att spendera tid och energi på repetitiva och enformiga uppgifter, kan fokusera på innovation, kreativitet och skapande – sådant robotar inte är kapabla till. [2]

I och med att specifikt RPA är en så pass ny teknologi är det begränsat med vetenskapliga artiklar och konferensmaterial. Det är i stället en del av nutiden och framtiden, med en enorm potential och mycket åtråvärdhet inom alla tänkbara arbetsområden. [2] Trenden visar på att det kommer ha en ännu större plats att fylla i framtiden; därför är det huvudsakliga syftet med detta projekt att utforska möjligheter, tillgängligheten och potentialen med denna teknologi.

1.2 Syfte och problemformulering

Syftet med detta projekt är att utforska möjligheter gällande övervakning med RPA, där övervakning refererar till någon form av kontinuerlig hämtning och rapportering av information från en digital källa. Arbetet kommer att spela på fördelen med RPA; nämligen att kunna jobba mot alla möjliga system och gränssnitt, och göra detta på ett nästintill mänskligt sätt.

Det som ska utredas är om uppgiften kan lösas på ett tillfredsställande och tillförlitligt sätt, alltså om den byggda roboten klarar av att hämta information från flera olika källor utan större problem, samt upprätthålla samma prestanda oavsett från var den hämtar informationen. Det ska också utredas var den byggda roboten misslyckas och vilka brister som kan hittas i teknologin.

1.3 Vetenskapliga mål

För att smalna av syftet och formulera mätbara mål har det valts att hämta information från olika väderwebbsidor, då dessa har olika gränssnitt och därmed kan demonstrera RPA:s fördel.

1. Kan en RPA-robot hämta likvärdig data från flera olika webbplatser (med samma funktion) med likvärdig prestanda?
2. Vilka brister och styrkor kan hittas med RPA-roboten?

1.4 Avgränsningar

Det valda testsetet för detta projekt kommer att vara RPA och datautvinning, där data specifikt kommer att hämtas från olika väderwebbsidor. Fokuset kommer också att vara på att jämföra prestandan för hämtningen på de olika webbplatserna samt göra olika tester för att se vilka brister som kan hittas.

Avgränsningarna utesluter annan funktionalitet inom RPA eftersom teknologin har så många användningsområden. Information kommer inte heller hämtas från någon annanstans, detta för att det ska vara möjligt att jämföra liknande information och fastställa om prestandan är likvärdig. Det enda verktyget som kommer att användas är UiPath då detta projekt inte går ut på att jämföra plattformar. Just UiPath väljs eftersom det är verktyget som uppdragsgivaren Knowit använder.

1.5 Översikt

Kapitel 2 beskriver den teoretiska bakgrunden för projektet, alltså teori som läsare bör vet för att förstå allting som nämns i rapporten. Det som beskrivs är RPA som teknologi, UiPath och UiPath Studio, samt hur det fungerar att bygga en RPA-robot i UiPath Studio.

Kapitel 3 beskriver den planerade metoden för projektet, alltså hur upplägget och planen ser ur för själva arbetet, samt planen för att besvara frågeställningarna.

Kapitel 4 beskriver den inledande studien för projektet. Där beskrivs kraven och vilka kriterier som har valts för att utföra uppgiften.

Kapitel 5 beskriver implementationen av den färdiga roboten, samt kort om hur mätresultaten ska behandlas.

Kapitel 6 beskriver resultatet – först den färdiga roboten, sedan mätresultaten från tidmätningarna och testerna som gjordes.

Kapitel 7 innehåller diskussionen för rapporten. Där diskuteras resultat, metod, vetenskaplig aspekt, konsekvensanalys och etiska och samhälleliga aspekter.

Kapitel 8, det sista kapitlet, innehåller slutsatsen för rapporten, där slutsatser dras av diskussionen, samt beskriver ett par möjliga framtida arbeten.

2 Teori

Den bakomliggande teorin i detta projekt är robotic process automation, vanligen förkortat RPA.

En annan stor del av projektet är UiPath och UiPath Studio, som är verktyget som kommer användas för att bygga RPA-roboten. Teorin kommer att täcka UiPath samt kort hur man bygger en RPA-robot i UiPath Studio.

2.1 Robotic Process Automation (RPA)

RPA har varit en växande trend sedan tidigt 2000-tal och har haft en ordentlig uppgång sedan sent 2010-tal. [1] Definitionen av RPA är att det är en framväxande teknologi med avancerade förmågor som genom mjukvarumekanismer tillåter automatisering av digitala processer tidigare utförde av människor. Detta innebär att RPA-roboten är en mjukvara som utför uppgifter som om en människa gjort det vid datorn, och att den är kapabel till mer avancerade uppgifter som sträcker sig över flera olika system och gränssnitt. [3]

Det finns många fördelar med RPA som har positiv inverkan på företag som har tagit sig an teknologin. RPA leder till kostnadsreduktion, ökad produktivitet och ökad tillfredsställelse bland anställda, detta tack vare att automatisering av processer frigör tid för anställda att lägga på mer produktiva och kreativa uppgifter. RPA ger också ökad noggrannhet och precision i och med att en robot alltid utför en uppgift på exakt samma sätt. När någonting blir fel har RPA granskningsmekanismer. Roboten är även mer tillgänglig än en person och gör inte intrång när den arbetar mot olika system och gränssnitt. RPA har också än låg teknisk tröskel vilket tillåter nästan vem som helst att bygga en robot. [3]

Med alla fördelar kommer en del utmaningar med RPA, där de flesta grundar sig i någon form av mänsklig faktor som okunskap och misstro. Där RPA används på arbetsplatsen är en utmaning de anställda som kan ha bristande kunskap och misstro för teknologin och hur den kommer påverka deras arbete. Ovana att implementera RPA kan leda till avsaknad av struktur, felaktig bedömning gällande vilka processer som kan automatiseras, samt generell undermålig implementation. När RPA väl är implementerad kräver den också hantering, som installation, underhåll, anpassning till komplexa och dynamiska miljöer, samt

undantagshantering där en människa måste interagera med roboten. Till sist kan enskild RPA, alltså RPA som inte kombineras med annan teknologi som AI, vara komplicerad att anpassa till komplexa och dynamiska miljöer. [3]

Det som krävs av en process för att automatisera den är, i generella drag, att den:

- är regelbaserad.
- har låg komplexitet.
- är väldigt repetitiv och utförs ofta.
- har inputvärden i digitalt format.
- är strukturerad/standardiserad (testad och verifierad så att det inte förekommer ändringar i framtiden) och dokumenterad (desto mer detaljerad processbeskrivning, desto mindre tid för implementation).
- interagerar med flera system. Denna punkt är inkluderad för att processen ska vara en optimal kandidat för RPA, vars stora fördel är att den kan arbeta över flera olika gränssnitt. [3]

Ett exempel på en undersökning som gjorts 2023 var på ett portugisiskt företag som använde sig flitigt av RPA. Resultatet visade att personer som jobbar med RPA rapporterar hög tillfredsställelse och finner RPA vara någonting väldigt positivt för deras arbete. Samtidigt var det få som fann någonting negativt – det var till exempel få som var rädda att RPA skulle ta över deras jobb. De anställda såg det som någonting positivt att RPA-robotar sköter enformiga och repetitiva digitala uppgifter eftersom det ger dem mer tid till kreativa och innovativa uppgifter. [2]

2.2 UiPath och UiPath Studio

UiPath är ett automationsföretag som startade 2005 i Rumänien. 2013 lanserade de en automationsprodukt som deras kunder kunde använda själva för att automatisera processer och 2019 hade de lanserat en RPA-plattform. [4] Med sin inlärningsplattform UiPath Academy nådde de 2021 över en miljon användare. Samma år hade de över 10 000 kunder världen över. [5]

UiPath Studio är en grafisk utvecklingsmiljö där man enkelt kan bygga RPA-robotar genom att välja, dra och släppa moduler som gör förutbestämda funktioner. Det involverar ingen direkt kodning, utan är enkelt att använda även för personer med låg teknisk kompetens. Det finns även UiPath StudioX som är en ännu mer användarvänlig version och är riktad till personer helt utan programmeringskunskap. I detta projekt är det Studio som ska användas. [6]

När man sedan har en färdigställd robot som ska köras används UiPath Assistant för att hantera den. UiPath Assistant är ett verktyg som används specifikt för att hantera färdiga robotar. Det är också Assistant som kör roboten när den startas från UiPath Studio. [7]

2.3 Att bygga en robot i UiPath Studio

UiPath Studio fungerar som många utvecklingsmiljöer – man öppnar ett projekt som man ska arbeta med på en arbetsyta. Den stora skillnaden är att det inte är ren programmering. Man använder sig i stället främst av redan skapade aktiviteter som både redan finns i Studio och som man kan lägga till med diverse bibliotek. För att bygga en robot använder man dessa aktiviteter, till exempel Use Application/Browser för att roboten ska starta en vald process på datorn. Vill man till exempel surfa in på smhi.se med Google Chrome öppnar man först webbsidan i webbläsaren. I UiPath Studio, i Use Application/Browser-aktiviteten, väljer man sedan 'Indicate application to automate', och klickar på webbläsaren med webbsidan. Man kan även manuellt skriva in rätt URL i aktiviteten. Detta arbetssätt är i princip detsamma genom hela processen att bygga en robot. Man väljer en aktivitet som man lägger till var man vill i programmet, sedan väljer man de element man vill att roboten ska interagera med. Utöver det kan roboten också utföra uppgifter i bakgrunden utan att interagera med någonting annat, som att göra uträkningar och hantera variabler. [8]

När det gäller att välja element använder sig UiPath av så kallade selektorer. Dessa selektorer säger åt roboten vilket element den ska interagera med, med hjälp av elementets attribut. UiPath använder sig då av den underliggande strukturen i applikationen för att peka på element. Det finns så kallade fuzzy och strikta selektorer, samt Computer Vision. Strikta selektorer pekar på ett enda element där alla valda attribut måste stämma, medan fuzzy inte är lika strikt och kan peka på flera liknande element. Computer Vision är en AI-modell som ser element

grafiskt. När man väljer ett element grafiskt ger UiPath automatiskt en selektor, oftast en fuzzy selektor med Computer Vision som extra stöd, samt så kallade ankare som är sekundära element med egna selektorer. Ankare agerar som stöd och gör det möjligt att hitta det önskade element även om fler element stämmer överens med selektorn. [9] En fuzzy selektor är inte lika strikt som en strikt selektor, utan tillåter viss förändring i ett elements attribut. Det är ett mer flexibelt sätt att peka på de önskade elementen och validerar med hjälp av flera attribut. [10]

UIExplorer är ett extra verktyg som listar upp hela trädstrukturen på applikationen man väljer. Man kan sedan leta igenom hela strukturen, markera valda element och hitta passande attribut för att skapa egna selektorer, både fuzzy och strikta. Man väljer då ett element, väljer vilka attribut som ska användas för att hitta den, båda av elementets egna och föräldraelements attribut, och sparar det som en selektor. [11]

2.4 Relaterat arbete

2.4.1 Cloth Consultant Robot with Temperature & Weather Report Using UiPath – RPA

Det här arbetet hade som mål att bygga en RPA-robot med UiPath som skulle hämta väderinformation från internet och sedan föreslå passande kläder till användaren. Resultatet var en robot som klarade av att hämta väderinformation som önskat, och meddelandet som visades på skärmen gav förslag på kläder beroende på vädret. Slutsatsen som gjordes var att det var snabbare för roboten att hämta informationen än för en människa att ta fram en väderwebbsida och hitta den själv. [12]

Deras arbete liknar detta projekt med hämtningen av väderinformation med en RPA-robot som byggdes i UiPath Studio. Det var dock ett arbete som endast fokuserade på att roboten skulle hämta väderinformation och sedan skriva ut ett meddelande med en rekommendation för kläder. Projektet i denna rapport har ett bredare fokus och inte riktigt samma målbild, där prestanda och jämförelse spelar in, samt vad som kan härledas från resultatet gällande RPA:ns roll i samhället.

2.4.2 Effectiveness of Robotic Process Automation for data mining using UiPath

I detta arbete var fokuset att testa hur effektiv en RPA-robot kan vara när den hämtar data från en webbsida. Även här användes UiPath Studio för att bygga roboten. Deras robot fungerade bra och de kom fram till att

RPA är bevisligen effektivt i fullbordandet av repetitiva uppgifter med minimala fel. Slutsatsen var också att RPA är väldigt anpassningsbart och skalbart, att det kan hantera komplexa uppgifter, och att med hjälp av verktyg som UiPath kan robotar byggas av personer utan programmeringserfarenhet. [13]

Likheter i deras arbete är att de använde sig av UiPath Studio för att bygga roboten och att målet var att testa hur effektiv en robot är vid hämtning av data. Likt det föregående relaterade arbetet var detta arbete på en något mindre skala än projektet i denna rapport med en smalare målbild. Det gjordes ingen jämförelse mellan olika webbsidor eller gränssnitt och fokuset låg helt på prestandan med den webbsida de testade roboten på. De drog inga större slutsatser kring RPA:ns plats i samhället, som tanken är med det här projektet.

3 Metod

Arbetet för detta projekt är uppdelat i två huvudsakliga faser som överlappar varandra – instudering och implementation. Planen är att instudering och implementation ska vara tidsmässigt jämnt fördelat över projektets gång. Detta är på grund av att instuderingen förmodligen kommer ta längst tid, medan själva bygget av roboten gissningsvis kommer ta två veckor. Att bygga en robot i UiPath Studio, även en mer komplex sådan, går relativt snabbt. Därför läggs mycket av fokuset på att lära sig om RPA, hur UiPath Studio fungerar, vad den är kapabel till och vilka begränsningar som finns innan det är dags att faktiskt bygga den slutgiltiga roboten.

3.1 Vetenskaplig metod

Detta projekt kommer främst att använda sig av kvantitativa metoder, då det är kvantitativt att mäta prestanda och bedöma tillförlitligheten hos roboten. För att hantera den första frågeställningen kommer data att sparas i en Excelfil, vilket sedan kommer kontrolleras för att se om roboten har hämtat rätt information. Tiden det tar för roboten att hämta data kommer också att mätas. Dessa två aspekter kommer göra det möjligt att besvara den första frågeställningen.

För att hantera andra frågeställningen kommer roboten att testas på diverse sätt, där problem medvetet kommer att skapas för att se vad roboten har för brister och styrkor. Resultaten från dessa tester kommer göra då göra det möjligt att besvara andra frågeställningen.

3.2 Arbetsmetod

Instuderingen, som klassas som teorin i detta fall, kommer huvudsakligen göras i UiPath Academy. Kurserna som kommer att genomföras i denna inlärningsplattform är främst relaterade till UiPath Studio, men också RPA som teknologi. Teorin kommer också bestå av att läsa tidigare skrifter gällande RPA för att fördjupa förståelsen, både för själva teknologin och dess roll i samhället och på arbetsmarknaden.

Den inledande studien består av design av roboten. Här kommer exakta väderwebbsidor att väljas. Robotens funktionalitet kommer att bestämmas efter att vädersidorna har blivit valda, eftersom de krävs för att bygget ska kunna påbörjas. Syftet med detta steg är att ha en

förutbestämd design innan implementationen, med utrymme för förändringar.

Implementationssteget består av att bygga roboten. Detta görs i UiPath Studio och kommer att använda sig av diverse väderwebbsidor för att samla information. Målet är att ha fullständig robot, eller robotar, som inte saknar någon planerad funktionalitet.

Förmodligen den enda konkreta mätningen som kommer att göras är hastigheten på hämtningen av information från de olika vädersidorna. Det kommer även kontrolleras huruvida roboten har hämtat den information som den skulle. Planen är att detta görs manuellt, men det är öppet för förändring. Att det görs så pass få mätningar beror på att det huvudsakliga målet är att utforska datautvinningsmöjligheter med RPA och se om en robot kan prestera på samma nivå över olika gränssnitt, samt att använda resultatet som grund för att diskutera syftet. Det finns i nuläget inga andra mätningar som är relevanta till just det som ska testas, men det kan komma att inkluderas fler.

Utvärderingen kommer att bestå av avläsning av resultat från mätningarna. Målet med utvärderingen är att bestämma om den slutgiltiga mjukvaran fungerar som tänkt, samt att ge underlag för att besvara på alla projektets frågeställningar.

3.3 Projektutvärderingsmetod

Först och främst kommer det bedömas om målen uppnåddes. Om inte frågeställningarna kan besvaras på att tillfredsställande sätt kan projektet knappast anses vara lyckad, utan då finns det betydliga brister i arbetet.

För att se om målet har uppnåtts kommer resultatet att analyseras. Utifrån resultatet kommer det vara möjligt att bestämma om frågeställningarna har blivit besvarade. Gällande det övergripande syftet med projektet kommer diskussionen och slutsatsen att spela större roll. Där kommer det baserat på resultatet diskuteras kring och dras slutsatser om projektet har strävat mot syftet på ett tillfredsställande sätt.

4 Inledande studie

Den inledande studien för detta projekt består av att reda ut kraven för att bygga en RPA-robot som är kapabel till att utföra uppgiften på ett tillfredsställande sätt. Detta är en inledande studie med en kravfångst eftersom det endast finns en ansats i detta projekt, som bara kommer att utföras på ett sätt eftersom uppgiften är så pass specifik.

4.1 Kravfångst

Kravfångsten består av konsultation med Knowit och design av roboten för att reda ut vad som behövs. För att bygga en robot som hämtar väderinformation från olika sidor behövs det först och främst en plattform för att kunna bygga den. Det krävs också väderwebbsidor.

4.1.1 Plattform – UiPath

För att kunna bygga en RPA-robot krävs någon form av plattform eller utvecklingsmiljö där bygget sker. Man behöver också kunna köra roboten för att överhuvudtaget kunna använda den.

Valet av plattform för att bygga och köra roboten var enkelt. UiPath är plattformen som Knowit använder för automation, vilket gjorde den till det självklara valet eftersom det är Knowit som är uppdragsgivare.

Det finns andra verktyg för att bygga RPA-robotar, som Nintex, Blue Prism och Robocorp, men dessa var alltså inte relevanta. Detta projekt går inte ut på att jämföra olika plattformar och därför var det inte nödvändigt att överväga. Med UiPath är det dessutom möjligt att få professionell hjälp av Knowit vid potentiella problem.

4.1.2 Process som kan automatiseras

För att bygga en robot som ska utföra en uppgift krävs det att den uppgiften är möjlig att automatisera. Uppgiften som har utformats är därför en enformig sådan som alltid kommer att se likadan ut – nämligen att gå in på tre olika vädersidor och hämta samma data vid varje körning. Denna process uppfyller majoriteten av de kraven som listas under kapitel 2.1, med undantaget att den inte är särskilt standardiserad eller dokumenterad.

4.1.3 Väderwebbsidor

Den specifika uppgiften som valdes var att hämta väderinformation från webbsidor, vilket innebär att det krävs ett antal väderwebbsidor för att kunna hämta data och jämföra prestanda. Väderwebbsidorna som ska användas är följande:

- smhi.se
- yr.no
- klart.se

De två förstnämnda valdes eftersom de gör sina egna mätningar. Flera andra svenska vädersidor som övervägdes hämtade information från smhi.se och yr.no, vilket gjorde att de valdes bort för att eliminera redundans.

Klart.se hämtar data från Foreca i Finland. [14]

4.2 Kravanalys

Som sagt valdes UiPath som plattform utan större övervägning eftersom det är plattformen som uppdragsgivaren använder. Det finns flera andra RPA-plattformar som skulle kunna användas och som kanske till och med kan anses vara bättre. Syftet med detta projekt är dock inte att jämföra plattformar eller ta reda på vilken plattform som är bäst inom vilket område. Syftet är inte heller att ta reda på vilken plattform som fungerar bäst för just den här uppgiften. Det är därför inte nödvändigt att överhuvudtaget analysera andra valmöjligheter. UiPath är en väl etablerad plattform som erbjuder allt som behövs för detta projekt och det har inte hittats några nackdelar med att använda den som skulle motivera att överväga en annan plattform.

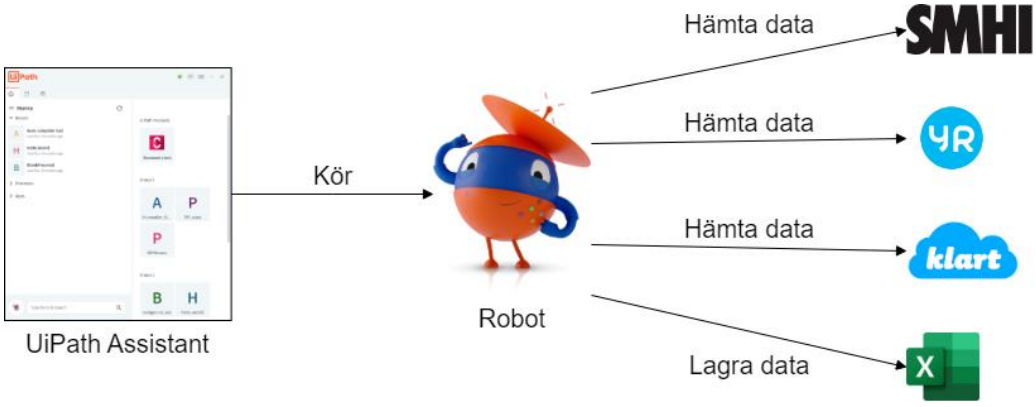
Att välja en process som kan automatiseras är inte lika klart. Det finns oändligt med processer som hade fungerat, kanske bättre än den valda processen, för ändamålet i projektet. Val av process gjordes därför efter visst övervägande och utefter förslag från Knowit, där förslagen bland annat var övervakning av elpriser och väder. Väder valdes eftersom det passade bättre till ändamålet att hämta likvärdig data från olika gränssnitt, alltså olika webbplatser som innehåller liknande information.

De tre sidorna som valdes gjordes efter beslutat att övervaka väder. Valet av väderwebbsidor var då relativt enkelt. Eftersom detta projekt görs i Sverige är det just svenskt väder som är intressant, och därför krävs det vädersidor som har svenska platser. När dessa skulle väljas övervägdes också var de fick sin väderdata från. Många sidor tar sin väderdata från andra sidor, och därför valdes smhi.se, yr.no, och klart.se, eftersom de tar väderdata från olika ställen.

I och med naturen av den inledande studien, där uppgiften är väldigt rättfram och det inte finns flera olika ansatser eller tillvägagångssätt, är också kravanalysen kort och rättfram. Det finns inte många jämförelser att göra, också på grund av att det finns oändligt med andra möjligheter, särskilt vad gäller vilken process som ska automatiseras. Det är helt enkelt inte möjligt att jämföra alla processer som skulle kunna automatiseras bara inom övervakning och datautvinning, därför smalnades det ner redan inför den inledande studien.

5 Implementation

Systemöversikten för detta projekt visas i figur 1. Hela systemet består av roboten i centrum, som körs av UiPath Assistant och interagerar med smhi.se, yr.no, klart.se och Excel.



Figur 1: Systemöversikt.

Detta kapitel kommer att gå igenom implementationen av systemet ovan. Det består av fem underrubriker som förklarar implementationen av de olika delarna av roboten. De tre första är implementationen för hämtning av data på de olika väderwebbsidorna. Eftersom de är så pass annorlunda från varandra har de olika utmaningar och kräver olika implementationer. Den fjärde underrubriken beskriver implementationen för lagringen av data i Excel, medan den femte beskriver övrig funktionalitet i roboten som inte täcks av de förgående punkterna.

5.6 beskriver hur allting ska mätas och utvärderas.

5.1 SMHI

Implementation för Smhi börjar med en Use Browser-aktivitet, där roboten startar en process av vald webbläsare (chrome.exe) och surfar in på webbplatsen med den angivna URL:en https://www.smhi.se.

Inom denna aktivitet är det först tre aktiviteter: Type Into, Click och en till Click. Den första används för att välja sökrutan och skriva in den användarvalda platsen. Den andra klickar på den valda platsen och den tredje klickar på länken till 10-dygnsprognosen. Den första och tredje av dessa aktiviteter har automatiskt valda selektorer – alltså selektorn är

den som UiPath Studio ger när man grafiskt väljer det elementet. De har också ankare som är valda på samma sätt. Den andra selektorn som väljer vald plats i listan är egengjord för att säkerställa att den väljer rätt element, se figur 2. Denna selektor letar efter ett element av typen Ul (Html unordered list-element) och med angivet aria-role och ID. I detta element kollar den efter ett element av typen Span (Html inline container) med index 1, samt kontrollerar att dess Innertext är samma som användaren angav. Detta gör att den väljer det första elementet som har rätt Innertext.

```
String.Format("<webctrl aria-role='listbox' id='downshift-0-menu' tag='UL' />
<webctrl tag='SPAN' idx='1' check:innerText='{0}' />", cityName)
```

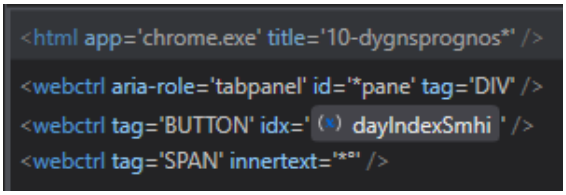
Figur 2: Exempel på selektor för Smhi, denna för att välja rätt plats vid sökning.

Följt av dessa går roboten in i en do-while-loop för datautvinning. Loopen har som villkor att en räknare (dayCounter, börjar på 0) ska vara mindre än antalet dagar användaren valt. Denna räknare tickas upp i slutet av loopen. I början av loopen finns en If-sats som kontrollera om räknaren är lika med 6. Är den det skrollar roboten ner på sidan till den sjunde dagen i listan, detta för att de sista dagarna ska synas.

För att hämta informationen används liknande selektorer, se figur 3. Dessa har inga ankare eftersom de är strikta och därmed inte har någon nytta av ankare. Smhi:s 10-dygnsprognos är uppbyggd som en tabellpanel av typen Div (Html division-element) med klickbara knappar av typen Button. Varje knapp är ett dygn och innehåller den nödvändiga informationen. Selektorn använder sig av tabellpanelen, knapparna och elementen av typen Span inom knapparna för att välja rätt element att hämta information från. Knapparna som representerar dygnen har ett index inom tabellpanelen som börjar på 2. Roboten använder en variabel dayIndexSmhi (med startvärde 2) som tickar upp i slutet av loopen.

I exemplet i figur 3 används Innertext för att hitta specifikt temperaturen bland alla element av typen Span inom typen Button. Roboten kollar då efter ett element av typen Span med en Innertext som stämmer överens med den angivna strängen (’*°’, asterisk representerar vilka och hur många tecken som helst), som ligger inom ett element av typen Button med index (idx) som är lika med dayIndexSmhi. Elementet av typen

Button måste också ligga inom tabellpanelen. Eftersom dayIndexSmhi tickar uppåt varje iteration kommer roboten att hämta information från de antal dygn framöver som användaren valt. Den rör sig alltså ner i listan.



Figur 3: Exempel på selektor för Smhi, denna för att hämta temperatur.

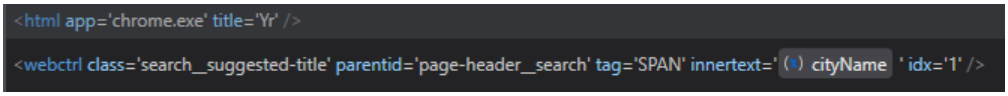
De andra selektorerna som hämtar prognosdatum, nederbörd, vindhastighet och väderlek är uppbygga på samma sätt, med endast små skillnader i målelementet. För prognosdatum letar den efter elementet av typen Span med index lika med 6. För nederbörd letar den efter Innertext lika med '*mm'. För vindhastighet letar den efter index lika med 10. För väderleken letar den efter ett element av typen Img med index lika med 1. Väderleken är en bild, men eftersom aktiviteten som används för att hämta alla information är Get Text hämtar roboten alternativtexten för bilden. Samtliga värden lagras i variabler och läggs i en datatabell varje iteration.

Se bilaga A för alla selektorer för Smhi.

5.2 Yr.no

Öppning av Yr.no i en webbläsare fungerar på samma sätt som för Smhi, bara med en annan URL (https://www.yr.no/nb).

De tre första aktiviteterna skiljer sig något – Click, Type Into och Click. Första Click klickar på sökknappen som öppnar sökfältet. Denna selektor är given av UiPath Studio, likaså ankaret. Detsamma gäller Type Into, som skriver in den valda platsen i sökfältet. Selektorn för andra Click är egengjord, se figur 4, för att säkerställa att den väljer rätt plats. Den tittar efter ett element med rätt Class, Parentid och Tag, detta för att den bara ska leta i dropdown-menyn vid sökning. Den kontrollerar också så att elementet har index lika med 1 och en Innertext som är detsamma som den valda platsen, för att den ska klicka på den första korrekta.

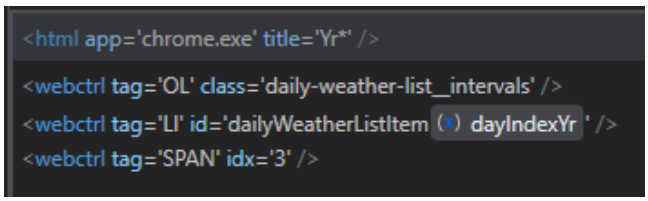


Figur 4: Exempel på selektor för Yr.no, denna för att välja rätt plats vid sökning.

I den följande do-while-loopen ligger först en If-sats som kontrollerar om räknaren dayCounter (nollställd mellan vädersidor), är lika med 6. Om så är fallet skrollar roboten när till det sjunde elementet i listan av dygn för att alla ska synas. Detta är samma funktionalitet som för Smhi, men med en helt annan selektor eftersom listorna av dagar är uppbyggda på olika sätt.

Yr.no har ett element av typ Ol (Html ordered list-element) och en specifik Class, se figur 5, som innehåller hela listan av tio dygn. Inom detta element finns element av typ Li (Html list-element), alla med ett ID som slutar på en siffra, där första elementet (första dagen) har ID 0. Med dayIndexYr, som börjar på 0 och tickas upp för varje iteration, kan roboten hitta rätt dag, alltså rätt element i listan.

Inom Li-elementen finns flera Span som innehåller den önskade informationen. I exemplet nedan letar den efter ett element av typen Span med index lika med 3 för att hämta temperaturen.



Figur 5: Exempel på selektor för Yr.no, denna för att hämta temperatur.

De andra selektorerna är uppbyggda på liknande sätt med några mindre skillnader. För att hämta prognosdatum letar den efter Ol och rätt Li-element, i vilket den letar elementet av typen Time. Detta räcker eftersom prognosdatum är det enda elementet av den typen inom Li-elementen.

För nederbörd pekar selektorn på ett element av typen Span inom Li-elementen med en Innertext lika med ’*mm’. Om det inte är någon

nederbörd för det dygnet är detta element dock tomt, vilket gör att variabeln för nederbörd kan bli utan värde. Därför är det en If-sats efter denna hämtning för att sätta variabeln lika med 0 om den är tom.

Yr.no visar fyra bilder för väderlekar, för natt, morgon, eftermiddag och kväll. Dessa fyra är Li-element inom Li-elementet för dygnet. Vid särskilda klockslag över dygnet töms dessa element för nuvarande dagens prognos – till exempel klockan 18:00 är bara väderleken för kväll kvar. För att hantera detta pekar denna selektor på det Li-element med index lika med 4, vilket är just väderleken för kväll. Detta garanterar att det alltid finns en väderlek för den nuvarande dagens prognos. Inom detta Li-element letar den efter elementet av typen Img för att få tag på alternativtexten för bilden.

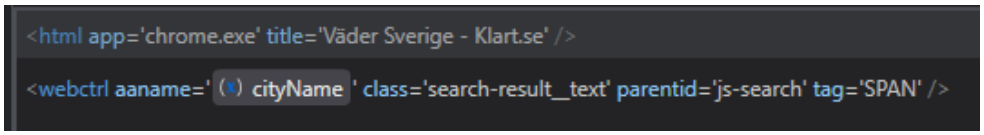
För vindhastighet letar den efter ett element av typen Span med Innertext lika med ’*m/s’ inom dygnets Li-element.

Se bilaga A för alla selektorer för Yr.no.

5.3 Klart.se

Öppning av Klart.se fungerar på samma sätt som de två andra, med URL:en https://www.klart.se/.

Det första som skiljer sig här är försöket att gå vidare från annonssidan som kommer upp om webbläsaren inte har webbplatsdata för Klart.se. Den första aktiviteten är då en Check App State-aktivitet som kollar statusen på elementet (med en strikt selektor) som är knappen att gå vidare till Klart.se. Om det elementet dyker upp kommer den att klicka på knappen med en Click-aktivitet, annars gör den ingenting. Följt av denna Check App State-aktivitet kommer en Type Into-aktivitet som väljer sökfältet och skriver in vald plats. Denna selektor och dess ankare är valda av UiPath. Click-aktiviteten för att välja rätt plats i listan har en egengjord selektor, se figur 6. Den letar efter ett element med ett specifikt ParentID för att bara kolla i resultatlistan, med en specifik Class, av typen Span, och med Aaname lika med den användarvalda platsen.

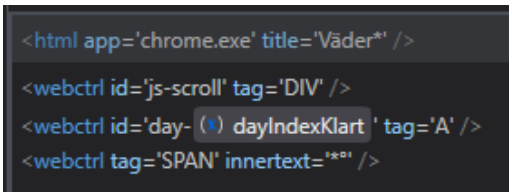


Figur 6: Exempel på selektor för Klart.se, denna för att välja rätt plats vid sökning.

Klart.se har en horisontell lista med 14 dygn som visar fem-sex dygn per spann, och två pilar för att röra sig fram och tillbaka. I den följande do-while-loopen är det först en If-sats som kontrollerar om räknaren dayCounter är lika med 6 eller 9. I de två fallen kommer roboten att trycka på högerpilen för att förflytta sig i listan och försäkra sig om att dygnen den letar efter är synliga.

Listan befinner sig inom en Div och består av A-element (Html hyperlink-element). Div-elementet är den skrollbara listan och A-elementen är dygnen. Alla A-element har ett ID med en sista siffra, där första dygnet har siffran 1, och så vidare.

Selektorn ser därmed ut som i figur 7, där den pekar på Div-elementen med det specifika ID:t för listan. I detta element pekar den på A-elementen med det ID som stämmer överens med dayIndexKlart, som har startvärde 1 och tickar upp för varje iteration. Inom detta element letar den efter ett element av typen Span med en Innertext lika med ’*°’ för att hitta temperaturen.



Figur 7: Exempel på selektor för Klart.se, denna för att hämta temperatur.

Selektorn för prognosdatum letar efter ett element av typen Time med index lika med 1, detta eftersom det finns fler element av samma typ.

För nederbörd letar den i stället för ett element av typen P (Html paragraph-element) inom A-elementet, med en Innertext lika med ’*mm*’. Den kräver asterisk både före och efter då Innertext

innehåller flera blanksteg före och efter själva värdet som måste räknas med för att roboten ska kunna hitta elementet.

För att hämta text för väderlek var det inte möjligt att använda en Get Text-aktivitet eftersom bilden inte hade någon alternativtext. Det används i stället en Get Attribute-aktivitet, som kan hämta valfritt attribut från det valda elementet. För väderlek letar roboten efter rätt element, vilket är av typ SVG (Scalable Vector Graphics, Html SVG image-element) och med index lika med 2 inom A-elementet. Detta element innehåller en aria-label-attribut som innehåller prognostexten, som Get Attribute-aktiviteten kan hämta och spara i en variabel.

Selektorn för vindhastighet ser nästintill exakt likadan ut som den för nederbörd, med den enda skillnaden att Innertext ska vara lika med ’*m/s*’.

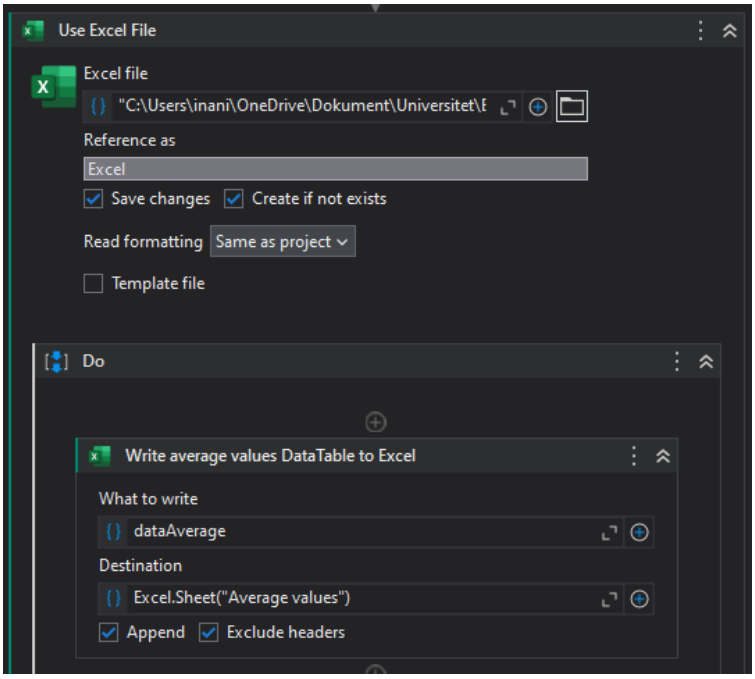
Se bilaga A för alla selektorer för Klart.se.

5.4 Excel

För att lagra alla data i Excel används en Excel Process Scope-aktivitet i slutet av automationen, som startar en Excel-process på datorn. Inom denna aktivitet öppnar roboten först den angivna Excel-filen för de enskilda värdena från de tre vädersidorna med en Use Excel File-aktivitet. I denna aktivitet anges filvägen till önskad Excel-file, men roboten kommer också skapa filen om den inte finns. Den sparar även alla ändringar den gör. Inom Use Excel File-aktiviteten finns tre Write DataTable to Excel-aktiviteter, en för varje datatabell med data från sin respektive vädersida. Inom dessa aktiviteter är det angivet vilken datatabell som ska skrivas och vilket blad i Excel-filen den ska skrivas till. Aktiviteterna är också konfigurerade att lägga till data utan att skriva över någonting och att exkludera datatabellernas headers.

Följt av denna Use Excel File är det ytterligare en sådan aktivitet som öppnar en annan Excel-fil, denna för att lagra alla genomsnittliga värden. Denna fungerar på samma sätt som den förgående, men har endast en Write DataTable to Excel-aktivitet, se figur 8. Den tar datatabellen med genomsnitten och skriver den till Excel-filens enda blad. Även denna aktivitet lägger till utan att skriva över och exkluderar datatabellens header.

Exempel på hur Excelbladen ser ut efter körning finns i bilaga C.



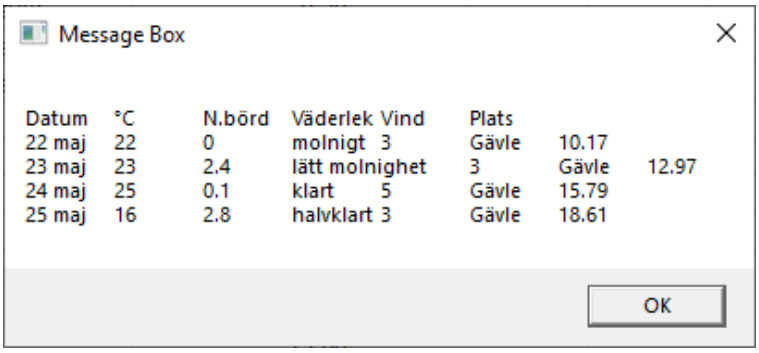
Figur 8: Exempel på Excel-aktiviteter, här en Use Excel File och en Write DataTable to Excel.

5.5 Övrig funktionalitet

5.5.1 Användarvänlighet

Det allra första roboten gör att fråga användaren om stad och antal dagar. Detta görs med dialogrutor som dyker upp på skärmen, både med en inputruta i vilka användaren kan skriva önskad plats och antar dagar att hämta information för. Båda dessa värden lagras sedan i variabler och används för att söka på stad och bestämma hur många dagar looparna ska gå över.

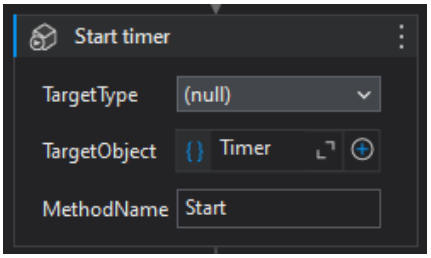
Det allra sista roboten gör är att skriva ut de införskaffade genomsnitten av värdena från vädersidorna i en meddelanderuta som dyker upp på skärmen, se figur 9. Informationen tas från datatabellen som innehåller de slutgiltiga genomsnitten som också skrivs till Excel-filen. Detta görs genom att ta all information från datatabellen och kopiera det till en enda sträng med en Output Data Table-aktivitet, som då kan städas upp och formateras för läsbarhet i meddelanderutan.



Figur 9: Exempel på meddelanderutan som dyker upp i slutet av körningen, här efter hämtning av data från fyra dagar med Gävle som plats.

5.5.2 Tidmätning

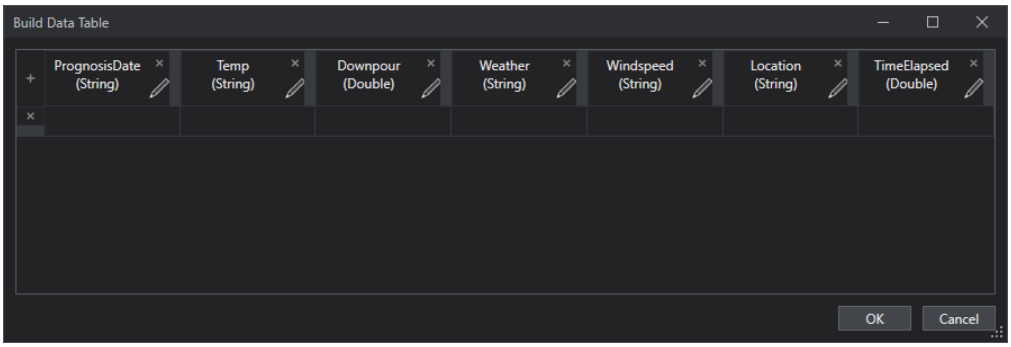
För att mäta tiden för de enskilda processerna används en förbyggd klass i UiPath Studio – Stopwatch. Ett objekt av denna typ har inbyggda metoder, varav tre används, samtliga i en Invoke Metod-aktivitet för att köras. Start används för att starta klockan efter att ett nytt Stopwatch-objekt instansierats, se figur 10. Inom alla tre datahämtningar från vädersidorna är en Start i början av loopen och en i slutet efter sista inhämtningen och innan datahanteringen. Innan Yr.no och Klart.se finns en Restart för att nollställa och starta klockan på nytt. Alla tidmätningar lagras i en variabel och sedan i vädersidornas respektive datatabell.



Figur 10: Exempel på en Invoke Method-aktivitet, här en Start med Stopwatch-objektet som TargetObject.

5.5.3 Datahantering

Datahanteringen går ut på att hantera och behandla all data som hämtas från vädersidorna. Innan varje Use Browser-aktivitet byggs datatabellen för den vädersidan med en Build Data Table-aktivitet. I denna aktivitet bestäms vilka kolumner datatabellen ska ha, vilka datatyper de ska vara och vilken datatabellvariabel den ska sparas i, se figur 11.



Figur 11: Kolumnerna för en datatabell, denna för Smhi. Alla datatabeller i programmet är identiska.

Inom varje loop för de tre sidorna läggs all data i variabler direkt i hämtningsaktiviteterna. Aktiviteten efter datautvinningen är en Assign som tar totala åtgångna tiden från Stopwatch-objektet, omvandlar den till den sträng och lagrar den i en variabel. Alla variabler behandlas med ett par metoder i en Multiple Assign-aktivitet – Replace för att ta bort oönskade tecken från strängen och Double.Parse för att omvandla ett par av strängarna (nederbörd och tidsåtgången) till Double för att möjliggöra uträkning av medelvärden. Följt av detta läggs de in i en datatabell specifik för den vädersidan med en Add Data Row-aktivitet. Detta utförs alltså varje iteration, vilket fyller datatabellerna med samtliga dagar som användaren önskade.

Efter att all data har hämtats byggs två datatabeller, en för de genomsnittliga värdena och en temporär för att kunna räkna ut medelvärden dynamiskt oberoende av hur många dagar användaren har valt.

Efter det är en while-loop, i vilken medelvärdena för de enskilda dagarna räknas ut. Den har villkoret att dayCounter (nu nollställd) ska vara mindre än antalet valda dagar. dayCounter används också för att välja rad i datatabellerna och tickar upp för varje iteration. Loopen innehåller tre If-satser som kontrollerar att de tre vädersidornas datatabeller varken är tomma eller att den nuvarande raden innehåller fel.

Inom varje If-sats, som bara utförs om datatabellen inte är tom och den nuvarande raden inte innehåller fel, är en Add Data Row-aktivitet, som kopierar raden till den temporära tabellen. Detta görs för alla tre väderdatatabeller så att den temporära endast innehåller data från en

enskild dag. Följt av If-satserna är en Multiple Assign-aktivitet. I denna beräknas medelvärdena på temperatur, nederbörd, vindhastighet och tidsåtgång. Detta görs genom att använda metoderna AsEnumarble (gör datatabellen räkningsbar) och Average (räknar ut medelvärdet) på en specifik kolumn i den temporära datatabellen, se nedan. Denna räknar ut medelvärdet på hela kolumnen Temp och returnerar en Double till en variabel:

```
placeholderTable.AsEnumerable().Average(Function(row)CDbl(row("Temp")))
```

Efter detta används Math.Round-metoden på alla fyra variabler för att runda av medelvärdena till två decimaler.

Nästa är en Add Data Row-aktivitet som lägger till dessa värden till datatabellen för genomsnittliga värden. Det som läggs till är prognosdatum från Smhi-raden, temperaturvärde, nederbördvärde, väderleken från Smhi-raden, vindhastighetvärde, platsnamn och tidsåtgångsvärde. Alla värden är alltså de uträknade medelvärdena.

Allra sist i loopen töms den temporära datatabellen för att den endast ska innehåll data för en enskild dag.

5.5.4 Felhantering

Det används främst Try-Catch för felhantering, med några If-satser för att fånga vissa utfall.

En Try-Catch är runt varje Use Browser-aktivitet för att hantera fallet att roboten inte klarar av att öppna sidan i webbläsaren. Vid fel skrivs ett meddelande att det inte var möjligt att komma åt sidan, sedan fortsätter programmet. Detta resulterar i en helt tom datatabell för den vädersidan.

Nästa Try-Catch är runt de tre första aktiviteterna för varje vädersida som söker efter och väljer önskad stad. Vid fel, till exempel om platsen inte finns med på den vädersidan, skrivs ett meddelande ut att platsen inte hittades. Detta resulterar i en helt tom datatabell för den vädersidan.

Det är sedan en Try-Catch inom varje vädersidas loop där roboten hämtar data, runt just datahämtningsaktiviteterna. Om någonting skulle misslyckas under den iterationen skulle hela den raden bli fel. Alla variabler sätts till ”Fel”, förutom nederbörd som är en Double och därmed sätts till 0.

If-satserna som hanterar vissa utfall har blivit beskrivna under tidigare underrubriker.

5.6 Mätuppställning

Hur tidmätningen fungerar i själva programmet har blivit beskrivet under 5.5.2.

Vid till exempel en datautvinning över fyra dagar fås det fyra tider i sekunder, både för samtliga tre vädersidor och för de genomsnittliga raderna, se tabell 1.

	Tid Smhi (s)	*Tid Yr.no (s)*	*Tid Klart.se (s)*	*Genomsnitt (s)*
Dag 1	11,92	7,09	11,51	10,17
Dag 2	14,74	9,83	14,34	12,97
Dag 3	17,59	12,59	17,20	15,79
Dag 4	20,42	15,38	20,02	18,61

Tabell 1: Tidsåtgångarna för alla tre vädersidor och genomsnitten av dessa tider, detta exempel över fyra dagar.

Tiden för dag 1 räknar med tiden det tar att öppna webbsidan, söka efter platsen, välja rätt plats och hämta data från första dagen i listan. Detta ger en bild över hur lång tid hela processen tar.

Tiderna för övriga dagar inkluderar tiden för hämtning från de föregående dagarna samt tiden för den dagen. Till exempel dag 3 inkluderar tiden att öppna webbsidan, söka efter platsen, välja rätt plats, och hämta data från dag 1, 2 och 3. Genom att ta skillnaden mellan dessa tider får man tiden för datautvinningen för just den dagen. På Yr.no tog det till exempel 2,76 sekunder att hämta data för dag 3.

Dessa tider används för att beräkna tidsåtgången för alla enskilda vädersidor (för öppning, sökning, val av rätt plats och datautvinning för en dag, samt för datautvinning för en dag) och genomsnitten på dessa tider. Med detta är det möjligt att få ett resultat gällande prestanda.

En annan viktig del av prestandan är hur ofta fel uppstår. Eftersom all data lagras i Excel-filer och med programmets felhantering är det möjligt att manuellt gå igenom all data och se om ett fel har uppstått. Alla körningar bevakas manuellt för att säkerställa att roboten inte gör ett fel som inte märks i Excel-filerna.

6 Resultat

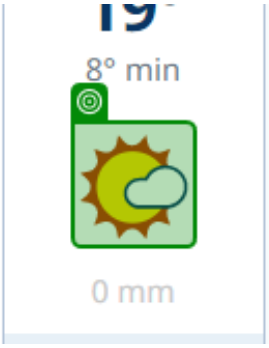
6.1 Färdiga applikationen

Den färdiga roboten i detta projekt är en robot som lever upp till kraven som ställts på den. Den klarar av att hämta information från tre olika väderwebbsidor, hantera all data och lagra det i Excel-filer.

I och med att detta inte är en applikation man lätt kan fånga på bild är det svårt att visa dess funktionalitet med endast bilder och beskrivning. Se därför bilaga B för en länk till en videodemonstration.

Något som kan visas med bilder är UiPath Studio som finner sina mål med de angivna selektorerna. Figur 12 och 13 är exempel på det.

Figur 12: UiPath Studio hittar ett mål på Smhi, här med en automatiskt skapad selektor och ankare.



Figur 13: UiPath Studio hittar ett mål på Klart.se, här med en egengjord strikt selektor.

6.2 Mätresultat

Se bilaga C för exempel på mätresultat.

Totalt gjordes 40 körningar av programmet i syfte att samla mätresultat och testa prestanda. Körningarna gjordes med olika inputs i början för plats och antal dagar, vilket resulterade i 148 rader data per vädersida, 444 rader totalt exklusive genomsnittsraderna också på 148.

Tabell 2 nedan visar sammanställningen av mätresultatet.

Testmiljö	*Datautvinning för en enskild dag (s)*	*Hela datautvinningsprocessen inklusive en enskild dag (s)*
Smhi	2,85	11,39
Yr.no	2,75	7,81
Klart.se	2,83	12,28
Genomsnitts-värden	2,81	10,47

Tabell 2: Genomsnittliga tidsåtgångar. Samtliga värden är i sekunder.

Detta visar på jämn prestanda över de tre vädersidorna gällande att hämta data för en enskild dag. För hela datautvinningsprocessen var det några sekunders skillnad. Anledningarna till det diskuteras i nästa kapitel. För att hämta data för en dag från alla tre sidor tar det i genomsnitt

$$11{,}39 + 7{,}81 + 12{,}28 = 31{,}48 \text{ sekunder.}$$

Även inom resultaten för de tre sidorna var tidsåtgångarna jämna, vilket visas med standardavvikelserna i tabell 3 nedan.

Testmiljö	*Datautvinning för en enskild dag (s)*	*Hela datautvinningsprocessen inklusive en enskild dag (s)*
Smhi	0,085	1,335
Yr.no	0,062	0,963
Klart.se	0,099	1,389

Tabell 3: Standardavvikelse för respektive vädersida och mätpunkt. Samtliga värden är i sekunder.

Vad gäller antalet fel som uppstod under körning skedde endast ett. Inga andra fel observerades under alla 40 körningar och inga andra fel dokumenterades i den data som sparades.

Felet som uppstod var under körning nummer 23 då Åre skrevs in som val av plats. Denna plats fanns inte med på Klart.se, vilket resulterade i att alla rader från Klart.se för den körningen fylldes med ”Fel” i stället. Detta räknades då inte med i genomsnitten för Klart.se och inte heller det totala genomsnittet. Genomsnittet för de tre dagarna är därmed beräknade av data från Smhi och Yr.no.

6.3 Tester av brister

Ett test som kördes var att ha webbläsaren på en mindre skärm under körning där målelement inte längre var synliga. Antalet dagar som valdes var 2. På Yr.no och Klart.se fungerade automationen som vanligt för båda dagarna. På Yr.no var samtliga målelement synliga på skärmen medan de var utanför på Klart.se. På Smhi fungerade sökningen och val av plats i söklistan, men roboten kunde inte hitta länken till 10-dygnsprognos då denna inte var synlig. Anledningar till detta diskuteras i nästa kapitel.

Ett till likadant test gjordes, denna gång med 10 antal dagar. Resultatet på Smhi blev detsamma – den kunde inte hitta 10-dygnsprognos. Yr.no fungerade som den skulle. På Klart.se när den kom till aktiviteten att klicka på pilen för att navigera i listan klickade den i stället på en annons, vilket tog den till en helt annan webbplats. Detta resulterade i att programmet kraschade då det blev fel i programmets datahantering.

Ett annat test som gjordes var att ändra URL:en för implementationen för Smhi till URL:en för Yr.no. Resultaten blev att den inte hittade någonting på Yr.no med Smhi:s selektorer. Programmet avslutades när den gick vidare från Smhi:s implementation eftersom ändamålet med testet var uppnått.

Nästa test var att manuellt minimera webbläsaren under hela körningen. Antalet valda dagar var 3. Vid aktiviteterna Use Browser, Click och Type Into öppnade/förstorade roboten webbläsaren av sig själv. När det skedde minimerades webbläsaren så snabbt som möjligt igen. På Smhi hade inte roboten nog med tid för att trycka på 10-dygnsprognos vilket resulterade i att tre rader blev fel. På Yr.no hann den trycka på vald plats i söklista och lyckades hämta all information. På Klart.se hann den också klicka på vald plats och kunde hämta all information.

7 Diskussion

7.1 Diskussion och analys av resultat

Den slutgiltiga roboten uppfyllde mestadels av det som önskades, och uppfyllde allt som krävdes av den. Det är ett par funktioner den skulle ha haft men som inte implementerades på grund av tidsbrist. En av dem är hantering av dagens väderlek på Yr.no. I nuläget tar roboten endast väderleken för kvällen på samtliga dygn. Önskemålet hade varit att den i stället haft en hantering, till exempel genom att kolla på klockslaget vid körning och välja väderlek baserat på det. Att endast ta väderleken för kvällen är kanske inte det mest optimala eller representativa för väderleken för dagen. Det mest intressanta hade förmodligen varit vädret för eftermiddagen då flest personer är ute och rör på sig.

En annan funktion hade varit att göra ett genomsnitt av väderförhållandena. I nuläget är det Smhi:s väderlek för dagen som läggs in i genomsnittstabellen. En idé för detta var att skapa en tabell där likvärdiga väderlekar från de tre vädersidorna skulle representera en siffra. Roboten skulle sedan under datahantering kunna omvandla väderlekarna till dessa siffror, beräkna medelvärdet, runda av till närmaste heltal och sedan omvandla tillbaka till den väderlek siffran representerar. Detta hade gett ett mer sammanställt resultat, som för de övriga värdena.

Allt som allt är dock roboten bra och fungerar bra för ändamålet, och den gör det snabbare än en människa med en total processtid på under en minut.

Vad gäller prestanda visade roboten på en väldigt jämn prestanda över de olika vädersidorna. Skillnaden mellan tidsåtgångarna för att hämta data för en enskild dag är försumlig och knappast märkbar inom den mänskliga upplevelsen. Den höll även väldigt jämn prestanda mellan varje datahämtning per dag med väldigt låga standardavvikelser för alla tre. Standardavvikelsen för Smhi var något högre än Yr.no, och Klart.se var något högre än Smhi. Detta beror på hur roboten hanterar val av antal dagar högre än sex. På Smhi och Yr.no skrollar den ner på sidan, medan den klickar på en knapp på Klart.se. Klart.se har störst standardavvikelse eftersom roboten måste trycka på knappen två gånger om antalet valda dagar är tio, vilket var fallet i en av körningarna. Skillnaderna beror också på hur webbsidorna är uppbyggda och hur lätt det är för roboten

att hitta målelementen. Även fast den endast skrolla ner en gång vid högre antal valda dagar på både Smhi och Yr.no har roboten något bättre prestanda på Yr.no. Utan några märkbara skillnader i implementationen är den rimliga förklaringen att Yr.no:s struktur är enklare för roboten att navigera och leta sig igenom.

Det är större skillnad när det kommer till tidsåtgången för hela processen inklusive datahämtning för en dag. Roboten är klart snabbast på Yr.no, och detta beror främst på implementationen och vädersidornas struktur. På Smhi måste roboten ta ett extra steg som inte krävs på de andra två sidorna – efter att ha valt rätt plats bland sökresultaten måste den navigera till Smhi:s 10-dygnsprognos genom att klicka en länk som leder till en annan sida. Denna sida måste dessutom laddas in. Detta adderar ett par sekunder på processen. På Klart.se dyker det upp en annonssida om webbläsaren saknar webbplatsdata. Robotens första aktivitet för Klart.se, att försöka klicka förbi annonssidan, lägger till tre sekunder på processtiden.

Även när man räknar med dessa två faktorer verkar roboten fortfarande ha bättre prestanda på Yr.no. Igen är den rimligaste förklaringen att Yr.no är enklare för roboten att navigera och det går snabbare för den att hitta alla målelement. Det beror förmodligen också på inladdningstiden för sidan, eftersom tidsåtgången räknas från att roboten startar Google Chrome-processen och surfar in på sidan. Inladdningstiden när den väljer plats i söklistan spelar också in. Sammantaget är det förmodligen inte roboten som har problem med prestanda eller liknande, utan det är vädersidorna som är så pass olika att robotens resultat blir olika.

Vad gäller hur många fel som uppstod under testkörningarna visar detta på att roboten är tillförlitlig. Det enda felet som uppstod berodde inte heller på roboten, utan var endast på grund av att Klart.se inte har Åre som ett alternativ. Roboten gjorde då exakt det den skulle – den markerade de raderna som fel och fortsatte med sin uppgift. Felhanteringen fungerade som den skulle. I och med att det enda felet som uppstod inte var något fel av roboten kan den hittills bedömas som ytterst tillförlitlig.

De övriga testerna som gjordes hade som syfte att medvetet skapa potentiella problem och se var roboten har brister och styrkor. Resultatet av dessa tester visar på att det finns brister i hur en RPA-robot fungerar.

Många problem kan också härledas till implementationen och vilka selektorer man valt.

En förklaring till varför roboten inte klarar av att hitta länken till 10-dygnsprognos på Smhi är att den selektorn är fuzzy och förlitar sig i andrahand på Computer Vision, samt har den ett ankare med en fuzzy selektor som inte heller syntes under testet. Roboten försöker alltså hitta elementet visuellt med viss hjälp av webbplatsens struktur. Detta är till skillnad från andra selektorer som är strikta och helt förlitar sig på webbplatsens struktur och därmed endast kan peka på ett enda element. Med strikta selektorer kan roboten hitta element även fast de inte syns och hanterar risken att roboten interagerar med något annat element. Därför skulle det vara fördelaktigt att använda strikta selektorer i största möjliga mån för att kringgå problemet att den inte kan hitta element som inte syns på skärmen.

En annan tydlig brist är att selektorer är väldigt specifika. Selektorer som fungerar på ett gränssnitt kommer inte fungera på ett annat om inte det gränssnittet har exakt samma struktur. Det kan också uppstå problem om strukturen på ett gränssnitt förändras. I det fallet kommer roboten plötsligt tappa bort sig och inte hitta sina målelement eftersom selektorn inte längre fungerar.

RPA har helt klart begränsningar. Den kan kombineras med andra teknologier som AI (artificiell intelligens) för att göra den smartare och för att hantera fler situationer, men ensamstående är det en ganska stel teknologi i att den kräver tydlig struktur och regler. Den är dock väldigt bra för sitt ändamål, nämligen att utföra enformiga, regelbaserade digitala uppgifter. Den har också sin stora fördel i att den jobbar ytligt och fungerar mot alla möjliga gränssnitt och system. För specifikt övervakning och datautvinning är den därför stark, eftersom den kan kontinuerligt ta in information, hämta data och varna vid förändringar, förutsatt att det den övervakar är regelbaserat och har en struktur roboten kan förlita sig på.

Det är också fortfarande en ny teknologi med stor potential som är under konstant utveckling. I kombination med andra teknologier kan den vara ännu starkare. En RPA-robot i kombination med AI skulle till exempel lätt kunna ta sig förbi ”Är du en robot?”-kontroller. En sådan robot skulle också vara ännu starkare inom övervakning och datautvinning eftersom

AI:n lättare skulle märka och hantera förändringar i data, samt hantera strukturförändringar. Trenden är fortsatt uppåtgående och RPA kommer förmodligen ha en allt större roll i framtiden, inom alla möjliga arbetsområden.

7.2 Metoddiskussion

I stora drag har metoden följts. De två planerade faserna av projektet, instuderingen och konstruktionen, tog upp ungefär den tid som planerat, med en förskjutning framåt på en vecka. De överlappade varandra som tänkt, där instudering fortfarande skedde när bygget hade påbörjats, eftersom det behövdes viss en pröva-sig-fram-metod för att bestämma exakt vad roboten skulle göra och vilka kunskaper som krävdes för att implementera det.

Teoridelen involverade inte lika mycket UiPath Academy som var tänkt. Mycket av de specifika kunskaperna som krävdes för implementationen togs i stället från UiPath Studio:s dokumentation och UiPath:s hjälpforum, där andra personer hade ställt liknande frågor och fått svar. Till väldigt stor hjälp var handledaren på Knowit som svarade på alla funderingar och hjälpte med problem som uppstod.

Den inledande studien fortgick i princip exakt som planerat. Det var ursprungligen en ganska lös plan med mycket utrymme för förändring, men det var möjligt att hålla sin inom de planerade ramarna. Den exakta designen för roboten bestämdes genom att testa sig fram inom ramarna för projektet.

Implementationen gick också mestadels som planerat. Den påbörjades lite senare än tänkt, men det hanns ändå med att bygga en färdig robot. Implementationen överlappade något med den inledande studien eftersom det föregående steget involverade att testa sig fram. Det byggdes därför en första version av roboten som endast hämtade information väderinformation för den nuvarande dagen. Fokuset med den första versionen var att sökningen och hämtningen av data skulle vara felfri. När det fungerade som det skulle kopierades version 1 över till version 2. Den andra versionen ändrades sedan för att ha all önskad funktionalitet – felhantering, fullständig datahantering med uträkning av genomsnitt, interaktion med användaren, möjlighet att kolla upp till tio dagar och skapandet av en sammanställning för de valda dagarna.

Det hanns tyvärr inte med att implementera allt utöver det nödvändiga, detta såklart på grund av att implementationen påbörjades för sent.

Den största mätningen som gjordes var just tidmätningen, som var enligt planen. Det kontrollerades också att roboten hämtade rätt information och hur många fel det blev under testkörningarna. Det som lades till var att testa robotens brister och begränsningar, som var ett senare tillskott i projektet eftersom det behövdes fler vinklar för att kunna föra en bra diskussion kring teknologin.

Sammantaget så har metodens faser varit framgångsrika. Allt har inte följts helt och hållet och vissa ändringar har gjorts, men metoden ska ändå anses vara lyckad med tanke på att ändringar alltid görs.

7.3 Vetenskaplig diskussion

Kunskapen som kan härledas från detta projekt är både specifik och generell. Specifik in den formen att vi kan bestämma att en RPA-robot är kapabel till att hämta likvärdig information över olika gränssnitt, och den kan göra det med likvärdig prestanda. Den kan också göra det tillförlitligt under korrekt omständigheter. Den generella kunskapen gäller mer var RPA:ns styrkor och svagheter ligger. Den är stark i att utföra uppgifter på samma sätt gång på gång. Men den är också svag i att förutsättningarna måste vara rätt. Stör man roboten kan den tappa bort sig. Implementerar man den inte bra nog kanske den inte hittar ett element bara för att det inte syns på skärmen.

Jämfört med de två relaterade arbetena som nämns i kapitel 2 har mer generell kunskap erhållits, vilket var tanken. Det första relaterade arbetet hade endast som slutsats att det var möjligt att bygga en robot för deras ändamål. Det andra arbetet hade som slutsats att en RPA-robot har bra prestanda när det kommer till att hämta information från en enda webbplats. Ingen av dem diskuterade djupgående kring RPA:ns potential eller framtid.

Rent vetenskapligt kan därför det här projektet anses ge en bra grund för att besvara, eller i alla fall diskutera mer generella och framtida frågor gällande RPA som teknologi. Här har RPA:ns styrkor och svagheter påvisats, både generellt och gällande datautvinning, och därför kan man föra en diskussion kring hur den skulle fungera inom både övervakning och andra användningsområden.

Det här projektet har dock fortfarande brister. Det är egentligen inte omfattande nog för att besvara de stora frågorna kring vad exakt RPA är kapabel till. Uppgiften som gjordes är inte heller komplicerad nog för att ge en djupare förståelse för teknologins potential och framtid. Att hämta väderinformation från olika vädersidor är en droppe i sjön som är RPA:s kapacitet.

7.4 Konsekvensanalys

Det här projektet bidrar med kunskap och perspektiv gällande RPA:ns styrkor och svagheter. Det bör bidra till en minskad oro för att använda RPA-robotar på arbetsplatsen – en robot är inte kapabel till att ta över en människas jobb. Den är helt enkelt inte smart nog.

Det bidrar också med diskussion kring RPA:s potential och framtid. På arbetsplatsen kommer den med största sannolikhet bara bli mer använd, vilket bör leda till ökad produktivitet och tillfredsställelse bland anställda. Detta projekt bör alltså uppmuntra till att använda RPA.

Det största bidraget är dock förmodligen att sätta RPA mer på kartan för studenter och universitet. Med en sådan uppåtgående trend och stora användningsområden kommer RPA som sagt bara att växa. Det är dessutom en väldigt intressant teknologi som är rolig att jobba med. Det största hoppet är därmed att det här projektet ska bidra och inspirera andra studenter att lära sig om och göra arbeten inom RPA.

7.5 Etiska och samhälleliga aspekter

Det här projektet i sig med just den här roboten har knappast någon etisk eller samhällelig påverkan då den endast hämtar väderinformation från olika väderwebbsidor och gör en sammanställning på detta. Även med projektets syfte i åtanke gällande övervakning blir svaret detsamma. Naturen av RPA gör att den inte gör några intrång, utan den jobbar ytligt utan att påverka. Den påverkar inte mer än om en person skulle titta på vädersidorna och skriva ner informationen på ett papper. I ett övervakningsscenario skulle inte roboten påverka mer än om en människa skulle övervaka manuellt.

RPA som teknologi har dock stor påverkan etiskt och samhälleligt, särskilt på arbetsplatsen. RPA-robotar används främst för att ta över enformiga och ofta upprepande digitala uppgifter tidigare utförda av människor. En stor oro med RPA är därför att personer kommer att

förlora sina jobb då de inte längre behövs – nu finns det digitala robotar som klarar av att göra det bättre och snabbare, som dessutom är mycket billigare i drift.

RPA:ns stora påverkan är alltså att den kanske kommer påverka människor personligen genom att ta över jobb och göra människor överflödiga för vissa arbetsuppgifter. Detta har dock i studier, som i exemplet under teorin, visats sig inte riktigt stämma. Att använda RPA-robotar på arbetsplatsen har i stället varit positivt för de anställda då de har mer tid till kreativa och innovativa uppgifter. RPA är som sagt en hyfsat stel teknologi och är enskilt inte smart nog för att hantera annat än regelbaserade processflöden. RPA kan även användas av privatpersoner för att automatisera saker i hemmet, och mycket tid och resurser kan sparas med användandet av robotar.

Mer oro kan komma in när RPA kombineras med andra teknologier. Till exempel AI i sig kanske inte har stor potential till att skapa problem. Men i kombination med RPA, som kan bete sig exakt som en människa obegränsad av gränssnitt och system, uppstår det etiska och samhälleliga frågor. En RPA-robot som en AI kontrollerar skulle kunna göra i princip vad som i den digitala världen – den skulle i teorin kunna göra allt en människa kan. Den skulle kunna övervaka allt den har tillgång till. Hur etiskt är det på arbetsplatsen, eller ute i samhället? Kommer digitala, smarta robotar att ta över mer än vad som är tänkt? Eller är människans intellekt och förmåga till nyanserat tänkande så pass ouppnåeligt att världen inte behöver oroa sig? I nuläget kan förmodligen bara framtiden besvara dessa frågor. Teknologin vi har i dag är inte tillräcklig för att helt ta över, men den utvecklas konstant. Inom en hyfsat snar framtid kanske ett kontor endast består av datorer som jobbar självständigt.

8 Slutsats

Detta projekt resulterade i en färdig RPA-robot som uppfyllde kraven som ställdes på den. Den saknade ett par önskade funktionaliteter, men sammantaget var bygget av en robot i UiPath Studio lyckat. Resultatet visade på att en RPA-robot är kapabel till att hämta likvärdig information från olika gränssnitt med likvärdig prestanda. Prestandaskillnaden som fanns berodde med största sannolikhet på väderwebbsidorna som användes, inte själva roboten. Roboten var även tillförlitlig och klarade samtliga testkörningar, där det enda felet som uppstod berodde på vädersidan.

Testerna som gjordes för att medvetet skapa problem för roboten visade på vissa begränsningar, där slutsatsen är att en RPA-robot är stark i rätt förutsättningar, men också känslig för situationer den är inte förberedd på, och för ändringar i förloppet eller gränssnittets struktur. Den är inte smart och flexibel, utan ämnar sig bäst för enformiga och regelbaserade uppgifter där förutsättningarna inte förändras.

Frågeställningarna och problemformuleringen kan därmed anses vara besvarade.

Baserat på resultaten som erhölls kunde det föras diskussioner kring RPA:ns potential och framtid. Slutsatsen där är att en RPA-robot är ytterst bra på datautvinning, och med dess kapacitet med säkerhet också ytterst bra på mer kontinuerlig övervakning av diverse data eller aktiviteter. En annan slutsats som också kan dras är att RPA endast kommer växa sig större, och bli starkare i kombination med andra teknologier.

Slutsatsen kring detta projekts påverkan är att dess största bidrag är kunskap gällande RPA, underlag för i vilka sammanhang RPA är starkt och svagt, och ett intresseskapande för teknologin. Slutsatsen är också att läsare av denna rapport ska vilja veta mer och att RPA ska ta en större plats inom utbildning med tanke på dess tillväxt och framtida plats i samhället, både på arbetsplatsen och i personers privatliv.

8.1 Framtida arbete

RPA har en otrolig potential och kan kombineras med andra teknologier, så framtida arbeten är närmare oändliga. Det är dock ett par konkreta framtida arbeten som har kommit upp under projektets gång.

8.1.1 RPA och AI

Ett starkt potentiellt framtida arbete är att kombinera RPA med AI. En sådan robot skulle i teorin kunna göra allt en människa kan utan begränsningarna som RPA har.

Ett sådant arbete skulle utföras på ett förmodligen liknande sätt som detta, men det skulle inte ha samma begränsningar gällande vilken process som skulle kunna automatiseras. Bara fantasi och tid skulle begränsa exakt vad projektet skulle handla om.

Det skulle dock involvera mycket mer instudering, både för RPA och AI, för att ha en bra förutsättning för en mer komplicerad robot som kan hantera komplexa uppgifter. Att köra ett sådant program skulle förmodligen också kräva ganska mycket datorkraft, eftersom den inte bara behöver köra en RPA-robot, utan också en AI-modell som på något sätt ska styra roboten.

8.1.2 Faktisk övervakning

Detta projekt fokuserade mer på datautvinning än ren övervakning. Roboten övervakar tekniskt sätt inte vädersidorna, utan den hämtar information när den körs. Ett framtida arbete, för att göra syftet till ett mål, skulle vara att bygga en riktig övervakningsrobot.

Denna robot skulle kontinuerligt övervaka valfri data, jämföra med tidigare data och varna vid förändringar eller när data når eventuella gränser. Den skulle till exempel kunna övervaka elpriser, logfiler, aktiviteter. Omfattningen på roboten skulle vara större vilket skulle kräva mer tid för implementation. Att köra en robot kontinuerligt skulle förmodligen också kräva ganska mycket datorkraft, eftersom den måste köras konstant under en längre tid med kontinuerliga kontroller på det som övervakas.

Referenser

[1] N. Ostdick, *"The Evolution of Robotic Process Automation (RPA): Past, Present, and Future,"* UiPath, 2016-06-26. Tillgänglig: https://www.uipath.com/blog/rpa/the-evolution-of-rpa-past-present-and-future [Hämtad 2023-05-01].

[2] M. Gomes, I. Seruca, "The perception of the management and the lower-level employees of the impacts of using Robotic Process Automation: the case of a shared services company," *Elsevier B.V.*, 2023.

[3] S. Moreira, H. S. Mamede, A. Santos, "Process automation using RPA – a literature study," *Elsevier B.V.*, 2023.

[4] UiPath, *"About us."* Tillgänglig: https://www.uipath.com/company/about-us [Hämtad 2023-05-03].

[5] UiPath, *"UiPath 2021 Impact Report,"* 2022-01-31. Tillgänglig: https://www.uipath.com/assets/downloads/2021-impact-report [Hämtad 2023-05-03].

[6] UiPath, *"Studio User Guide – Introduction."* Tillgänglig: https://docs.uipath.com/studio/standalone/2023.4/user-guide/introduction [Hämtad 2023-05-03].

[7] UiPath, *"About UiPath Assistant."* Tillgänglig: https://docs.uipath.com/robot/standalone/2023.4/user-guide/about-uipath-assistant [Hämtad 2023-05-15].

[8] UiPath, *"Creating a Basic Process."* Tillgänglig: https://docs.uipath.com/studio/standalone/2023.4/user-guide/creating-basic-process [Hämtad 2023-06-11]

[9] UiPath, *"About Selectors."* Tillgänglig: https://docs.uipath.com/studio/standalone/2023.4/user-guide/about-selectors [Hämtad 2023-06-11]

[10] UiPath, *"Fuzzy Search."* Tillgänglig: https://docs.uipath.com/studio/standalone/2023.4/user-guide/fuzzy-search-capabilities [Hämtad 2023-06-11]

[11] UiPath, *"UIExplorer."* Tillgänglig: https://docs.uipath.com/studio/standalone/2023.4/user-guide/uipath-explorer [Hämtad 2023-06-11]

[12] S. Sujatha, S. H. Prasanth, "Cloth Consultant Robot with Temperature & Weather Report Using UiPath – RPA," 2021.

[13] Y. Ketkar, S. Gawade, "Effectiveness of Robotic Process Automation for data mining using UiPath," 2021.

[14] Klart.se, *"Om oss."* Tillgänglig: https://www.klart.se/om-oss/ [Hämtad 2023-05-05].

Bilaga A – Selektorer

Samtliga selektorer för alla tre vädersidor – Smhi, Yr.no och Klart.se.

https://drive.google.com/file/d/1_URsha3EAi99zw0Dnt1zXTLWMwmyH2Nt/view?usp=sharing

Bilaga B – Videodemonstration färdig robot

https://youtu.be/r9ThDl30gNA

Bilaga C – Exempel mätresultat

Smhi

Prognosis for	Temp (celsius)	Downpour (mm)	Weather	Windspeed (m/s)	Location	Fetch time
21 maj	20	0	klart	3	Sundsvall	11,997185
22 maj	15	0	lätt molnighet	3	Sundsvall	14,7578116
23 maj	18	0	halvklart	3	Sundsvall	17,5891775
24 maj	22	0	mulet	4	Sundsvall	20,4577822
21 maj	23	0	klart	2	Gävle	10,0155399
22 maj	22	0	molnigt	2	Gävle	12,8412506
23 maj	22	5	halvklart	3	Gävle	15,705585
21 maj	20	0	klart	3	Stockholm	10,8944114
22 maj	22	0	halvklart	2	Stockholm	13,6717677
23 maj	20	0	molnigt	4	Stockholm	16,5185504
21 maj	21	0	klart	2	Uppsala	10,1221283
22 maj	23	0	halvklart	2	Uppsala	12,8448376
23 maj	20	0	molnigt	4	Uppsala	15,7006749
21 maj	14	0	lätt molnighet	5	Luleå	10,734695
22 maj	19	0	halvklart	3	Luleå	13,4877358
23 maj	11	6,5	regn	2	Luleå	16,3243418
24 maj	17	1,6	mulet	5	Luleå	19,206636
21 maj	21	0	klart	1	Sollefteå	10,9152537
22 maj	21	0	halvklart	3	Sollefteå	13,7771533
21 maj	21	0	mulet	4	Malmö	10,8780511
22 maj	26	0	klart	5	Malmö	13,6351464
23 maj	21	0,5	mulet	4	Malmö	16,4315273
21 maj	23	0	lätt molnighet	4	Göteborg	10,784577
22 maj	26	0	halvklart	3	Göteborg	13,6291353
23 maj	19	6,4	lätt regn	2	Göteborg	16,4690521
24 maj	16	0	klart	4	Göteborg	19,3215676
21 maj	21	0	klart	3	Mora	11,0810748
22 maj	22	0	halvklart	4	Mora	13,8697688
23 maj	17	13,4	mulet	3	Mora	16,671649
21 maj	20	0	lätt molnighet	1	Rättvik	10,7170597
22 maj	21	0	halvklart	2	Rättvik	13,4929257
23 maj	17	6,3	mulet	4	Rättvik	16,3458581
24 maj	20	0	halvklart	6	Rättvik	19,2720623
21 maj	11	0	klart	4	Umeå	10,8991284
22 maj	19	0	halvklart	3	Umeå	13,6644044
23 maj	18	0	molnigt	1	Umeå	16,5045441

Yr.no

Prognosis for	Temp (celsius)	Downpour (mm)	Weather	Windspeed (m/s)	Location	Fetch time
I dag 21. mai	21	0	klarvær	4	Sundsvall	7,4
Mandag 22. mai	16	0	delvis skyet	3	Sundsvall	10,15
Tirsdag 23. mai	18	1,1	skyet	4	Sundsvall	12,94
Onsdag 24. mai	22	0	skyet	3	Sundsvall	15,69
I dag 21. mai	24	0	klarvær	3	Gävle	7,15
Mandag 22. mai	22	0	skyet	3	Gävle	9,88
Tirsdag 23. mai	21	3,7	skyet	3	Gävle	12,59
I dag 21. mai	22	0	delvis skyet	3	Stockholm	7,45
Mandag 22. mai	23	0	delvis skyet	2	Stockholm	10,23
Tirsdag 23. mai	21	0	skyet	4	Stockholm	12,95
I dag 21. mai	23	0	klarvær	3	Uppsala	7,31
Mandag 22. mai	24	0	skyet	3	Uppsala	10,07
Tirsdag 23. mai	22	0	skyet	3	Uppsala	12,77
I dag 21. mai	15	0	lettskyet	5	Luleå	7,2
Mandag 22. mai	20	0	delvis skyet	3	Luleå	10,04
Tirsdag 23. mai	14	0	skyet	3	Luleå	12,77
Onsdag 24. mai	20	0	skyet	5	Luleå	15,51
I dag 21. mai	25	0	klarvær	2	Sollefteå	7,36
Mandag 22. mai	22	0	delvis skyet	3	Sollefteå	10,08
I dag 21. mai	24	0	skyet	5	Malmö	7,58
Mandag 22. mai	25	0	delvis skyet	5	Malmö	10,33
Tirsdag 23. mai	20	0	skyet	5	Malmö	13,14
I dag 21. mai	25	0	lettskyet	4	Göteborg	7,25
Mandag 22. mai	26	0	delvis skyet	4	Göteborg	9,98
Tirsdag 23. mai	18	0	delvis skyet	3	Göteborg	12,7
Onsdag 24. mai	20	0	klarvær	4	Göteborg	15,42
I dag 21. mai	24	0	delvis skyet	3	Mora	7,18
Mandag 22. mai	26	0	klarvær	4	Mora	9,94
Tirsdag 23. mai	28	1,3	regnbyger	4	Mora	12,68
I dag 21. mai	23	0	lettskyet	2	Rättvik	7,01
Mandag 22. mai	22	0	delvis skyet	2	Rättvik	9,75
Tirsdag 23. mai	16	3,1	skyet	2	Rättvik	12,51
Onsdag 24. mai	23	0	delvis skyet	5	Rättvik	15,23
I dag 21. mai	14	0	klarvær	4	Umeå	6,91
Mandag 22. mai	19	0	delvis skyet	3	Umeå	9,65
Tirsdag 23. mai	18	0	delvis skyet	3	Umeå	12,4
I dag 21. mai	17	0	lettskyet	6	Kiruna	7,31
Mandag 22. mai	18	0	klarvær	4	Kiruna	10,08
Tirsdag 23. mai	18	1	skyet	6	Kiruna	12,81

Klart.se

Prognosis for	Temp (celsius)	Downpour (mm)	Weather	Windspeed (m/s)	Location	Fetch time
21 maj	22	0	Klart	4	Sundsvall	11,81
22 maj	18	0	Nästan klart	3	Sundsvall	14,62
23 maj	19	1,5	Molnigt och regnskurar	3	Sundsvall	17,4
24 maj	22	0	Molnigt	4	Sundsvall	20,22
21 maj	23	0	Klart	3	Gävle	11,81
22 maj	21	0,1	Molnigt och lätt regn	3	Gävle	14,69
23 maj	21	1,6	Molnigt och regnskurar	3	Gävle	17,51
21 maj	22	0	Halvklart	3	Stockholm	11,83
22 maj	22	0	Halvklart	3	Stockholm	14,63
23 maj	21	0	Molnigt	5	Stockholm	17,42
21 maj	22	0	Halvklart	3	Uppsala	12,13
22 maj	22	0	Molnigt	1	Uppsala	14,96
23 maj	21	0,2	Molnigt	5	Uppsala	17,75
21 maj	14	0	Klart	5	Luleå	11,19
22 maj	16	0	Nästan klart	3	Luleå	14,04
23 maj	15	0	Molnigt	3	Luleå	16,84
24 maj	17	1,8	Molnigt och regnskurar	6	Luleå	19,61
21 maj	23	0	Klart	4	Sollefteå	11,56
22 maj	22	0	Nästan klart	4	Sollefteå	14,45
21 maj	24	0	Halvklart	4	Malmö	12,73
22 maj	21	0	Molnigt	6	Malmö	15,58
23 maj	18	1,5	Mulet och regnskurar	3	Malmö	18,4
21 maj	23	0	Klart	4	Göteborg	11,48
22 maj	23	0	Halvklart	3	Göteborg	14,28
23 maj	16	5,1	Mulet och regn	2	Göteborg	17,09
24 maj	18	0,3	Halvklart	3	Göteborg	19,9
21 maj	22	0	Nästan klart	3	Mora	12,1
22 maj	21	0	Halvklart	2	Mora	15
23 maj	16	5,4	Mulet och regn	2	Mora	17,87
21 maj	22	0	Nästan klart	2	Rättvik	11,31
22 maj	20	0,1	Halvklart	2	Rättvik	14,1
23 maj	17	1,5	Molnigt och lätt regn	3	Rättvik	16,9
24 maj	20	0	Molnigt	6	Rättvik	19,69
21 maj	16	0	Klart	4	Umeå	11,36
22 maj	19	0	Halvklart	2	Umeå	14,17
23 maj	19	0	Halvklart	3	Umeå	16,93
21 maj	17	0	Nästan klart	5	Kiruna	11,16
22 maj	17	0	Halvklart	4	Kiruna	13,95
23 maj	17	0,9	Molnigt och regnskurar	5	Kiruna	16,73

Genomsnitt

Prognosis for	Temp (celsius)	Downpour (mm)	Weather	Windspeed (m/s)	Location	Average time (s)
21 maj	21	0	klart	4	Sundsvall	10,4
22 maj	16	0	lätt molnighet	3	Sundsvall	13,18
23 maj	18	0,9	halvklart	3	Sundsvall	15,98
24 maj	22	0	mulet	4	Sundsvall	18,79
21 maj	23	0	klart	3	Gävle	9,66
22 maj	22	0	molnigt	3	Gävle	12,47
23 maj	21	3,4	halvklart	3	Gävle	15,27
21 maj	21	0	klart	3	Stockholm	10,06
22 maj	22	0	halvklart	2	Stockholm	12,84
23 maj	21	0	molnigt	4	Stockholm	15,63
21 maj	22	0	klart	3	Uppsala	9,85
22 maj	23	0	halvklart	2	Uppsala	12,62
23 maj	21	0,1	molnigt	4	Uppsala	15,41
21 maj	14	0	lätt molnighet	5	Luleå	9,71
22 maj	18	0	halvklart	3	Luleå	12,52
23 maj	13	2,2	regn	3	Luleå	15,31
24 maj	18	1,1	mulet	5	Luleå	18,11
21 maj	23	0	klart	2	Sollefteå	9,95
22 maj	22	0	halvklart	3	Sollefteå	12,77
21 maj	23	0	mulet	4	Malmö	10,4
22 maj	24	0	klart	5	Malmö	13,18
23 maj	20	0,7	mulet	4	Malmö	15,99
21 maj	24	0	lätt molnighet	4	Göteborg	9,84
22 maj	25	0	halvklart	3	Göteborg	12,63
23 maj	18	3,8	lätt regn	2	Göteborg	15,42
24 maj	18	0,1	klart	4	Göteborg	18,21
21 maj	22	0	klart	3	Mora	10,12
22 maj	23	0	halvklart	3	Mora	12,94
23 maj	20	6,7	mulet	3	Mora	15,74
21 maj	22	0	lätt molnighet	2	Rättvik	9,68
22 maj	21	0	halvklart	2	Rättvik	12,45
23 maj	17	3,6	mulet	3	Rättvik	15,25
24 maj	21	0	halvklart	6	Rättvik	18,06
21 maj	14	0	klart	4	Umeå	9,72
22 maj	19	0	halvklart	3	Umeå	12,49
23 maj	18	0	molnigt	2	Umeå	15,28