UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

Robotização de tarefas repetitivas e críticas nos centros de operação de ciber segurança

Sara Vanessa Sanches Lima do Nascimento

Mestrado em Engenharia Informática
Especialização em Arquiteturas, Sistemas e Redes de Computadores

Trabalho de Projeto Orientado por:
Profª. Doutora Ana Luísa do Carmo Correia Respício
Engº José António dos Santos Alegria

2019

Agradecimentos

Em primeiro lugar quero agradecer à Professora Ana Luísa Respício por ter aceite fazer este trabalho, por todo o acompanhamento, todos os conselhos, orientação e disponibilidade ao longo do ano. Ao Engenheiro José Alegria quero agradecer pela oportunidade, pelo acompanhamento, orientação e por reconhecer as minhas capacidades na realização deste trabalho. Agradeço à equipa DCY que me receberam de braços abertos dando-me o apoio incondicional, motivação e boa disposição. Obrigada a todos.

Às amizades feitas neste ano, quero agradecer ao Pedro Santos, ao Gonçalo Miranda, ao Fábio Fernandes, à Mariana Pina, ao Gonçalo Lima, ao André Sales e ao Maurício Carvalho por todo o apoio e auxílio. Também quero agradecer-vos por todas as conversas e risadas que me animavam motivando-me a continuar este percurso. A vocês, muito obrigada.

Quero agradecer á minha família, em especial á minha mãe Maria de Fátima que me deu um suporte estonteante durante a minha vida. Ao meu pai Rui por toda a ajuda, apoio, paciência, disponibilidade e companheirismo ao longo de todo este percurso. Quero agradecer ao meu irmão Daniel por todas as vezes que se prontificou a ouvir-me e continuo a dizer que és o melhor que tenho e foi a melhor prenda que podia ter recebido, obrigada pelos 21 anos de paciência e irmandade. Quero agradecer às minhas avós, Cristina e Deolinda por todo o apoio prestado e por toda a vossa ajuda que me fez chegar até aqui. Por fim, também quero agradecer aos meus tios Mário e Carla por todas as risadas, todos os conselhos e pelo carinho dado desde sempre.

Quero agradecer em especial ao Sasha Fonseca por todo o apoio que me foi dado ao longo de quatro anos. Levantaste-me sempre que caí e soubeste sempre o que dizer nos momentos certos. Acompanhaste-me grande parte do meu percurso académico e acreditas-te sempre em mim e naquilo que era capaz de fazer. Fizeste e fazes-me rir e sorrir constantemente dando-me a confiança que o amanhã vai ser sempre melhor. A ti, um obrigada não basta, mas espero mostrar o meu agradecimento todos os dias dando-te o melhor.

Quero agradecer a todos do ano 1314 por todas as aventuras ao longo do percurso académico, em especial aos meus irmãos DeRastos, Avante e Morena por todas as vezes que ouviram os meus desabafos e pelas vezes que me levantaram; também quero agradecer aos meus Padrinhos GPS e Bainha por todas as conversas, pelo suporte (não só académico mas também emocional) e pelas lições que me foram passadas e me fizeram crescer. Hoje parte do que sou, devo a vocês; quero agradecer ao ano 1516, mas em especial ao Imperial, Siga, Doidão, Tenerife e Dolores por todos os momentos passados e todas as brincadeiras. Vocês os cinco, ensinaram-me tanto e mostraram-me tanto. A todos, muito obrigada mesmo por tudo e levo cada um de vocês no meu coração.

Por fim, quero agradecer às amizades em Elvas, em especial ao Miguel Camões, à Isabel Santos e ao Filipe Pereira por todas as vezes que ouviram, por todas as vezes que se lembraram de mim perguntando quando voltava à "terrinha" e por todas as gargalhadas. Levo-vos sempre comigo, obrigada por tudo.

Aos meus pais, ao meu irmão e às minhas avós.

Resumo

A gestão de processos rotineiros envolvendo um grande número de tarefas repetitivas garantindo um elevado desempenho e um número de erros reduzido é um problema com o qual se deparam muitas empresas e organizações. Atualmente, os colaboradores efetuam estas tarefas rotineiras com as tecnologias fornecidas pela empresa, mas estão a chegar a um *bottleneck* na concretização das mesmas. A repetibilidade impacta do desempenho dos colaboradores pois o cansaço aumenta e o interesse e o agrado diminuem com o passar do tempo. Consequências como estas podem apresentar, futuramente, resultados negativos à empresa como a baixa na produtividade.

Estas consequências causadas pela repetibilidade dos processos levaram ao novo conceito *Robotic Process Automation* (RPA) que consiste no desenvolvimento de robots, baseados em regras, com capacidade para realizar algumas dessas tarefas repetitivas. Outro conceito (centrado na gestão de tarefas de cibersegurança) chamado *Security Orchestration Automation and Response* (SOAR) também se fundamenta na automatização de processos, contudo, foca-se em receber alertas vindos dos *Security Information and Event Management* (SIEM), filtrar os falsos positivos e mitigar os verdadeiros positivos a partir de regras estabelecidas pela equipa *Security Operation Center* (SOC). Estas soluções conduzem ao aumento do desempenho na realização das tarefas e aumentam a disponibilidade dos colaboradores para realizarem outras tarefas de maior importância que requerem uma análise mais detalhada.

Este relatório apresenta o trabalho realizado no contexto de um projeto de automação de processos de negócio no ambiente da Altice Portugal (MEO) no Departamento de *Cyber Security & Privacy*.

O objetivo deste trabalho foi explorar os conceitos RPA e SOAR, e implementar os casos de uso propostos pela equipa SOC da MEO. Após a implementação, foi feita uma análise empírica e teórica para comprovar que a inserção destes conceitos habilita aos analistas SOC a ter mais disponibilidade para concretizar processos de maior relevância. As tecnologias de RPA exploradas foram *Automation Anywhere Enterprise* e *Blue Prism*, e de SOAR foi o *ATAR Labs*.

Palavras-chave: desenvolvimento, automação, análise, tarefas, segurança

Abstract

Routine process management involving a large number of repetitive tasks ensuring high performance and a small number of errors is a problem many companies and organizations face. Currently, employees perform these routine tasks with the technologies provided by the company, but are reaching a bottleneck in the realization of them. Repeatability impacts employee performance as tiredness increases and interest and pleasure diminish over time. Consequences such as these may in future lead to negative business outcomes such as lower productivity.

These consequences of process repeatability have led to the new Robotic Process Automation (RPA) concept of developing rule-based robots that can perform some of these repetitive tasks. Another concept (focused on cybersecurity task management) called Security Orchestration Automation and Response (SOAR) is also based on process automation, however, it focuses on receiving alerts from Security Information and Event Management (SIEM), filtering false positives and mitigate true positives from rules set by the Security Operation Center (SOC) team. These solutions lead to increased task performance and increase staff readiness to perform other major tasks that require more detailed analysis.

This report presents the work done in the context of a business process automation project in the Altice Portugal (MEO) environment in the Cyber Security & Privacy Department.

The aim of this work was to explore the RPA and SOAR concepts, implement the use cases proposed by the MEO's SOC team. After implementation, an empirical and theoretical analysis was made to prove that the insertion of these concepts enables SOC analysts to be more available to carry out processes of greater relevance. The RPA technologies explored were Automation Anywhere Enterprise and Blue Prism, and SOAR was ATAR Labs.

Keywords: development, automation, analysis, tasks, security

Índice

Lista de Figuras

Lista de Tabelas

Capítulo 1
Introdução

1.1 Motivação

O propósito final da tecnologia é melhorar a qualidade de vida do Homem auxiliando-o nas tarefas diárias. Com o seu desenvolvimento, a tecnologia passou a ser uma necessidade diária e também um fator envolvente – propositadamente ou não – na vida do Homem.

Dentro do ambiente empresarial, a tecnologia facilita os trabalhadores a atingirem os seus objetivos diariamente (*i.e.*, ajuda-os a realizar rapidamente as tarefas pelas quais são responsáveis). Contudo, algumas das tarefas realizadas podem tomar grandes dimensões e dificultam a execução eficiente e fiável das mesmas. Além disso, há tarefas que apesar de serem necessárias para o bom funcionamento do ambiente laboral, não possuem um nível de complexidade elevado [1], o que implica que o responsável pelas mesmas não usufrui desse tempo em tarefas mais complexas e críticas. Atualmente, a automação de tarefas é feita através de *scripts*[1] mas o desenvolvimento destes é complexo porque todos os elementos integrados na execução das tarefas têm que estar bem definidos. Por exemplo, quando um botão numa página *web* é carregado, este botão possui um conjunto de identificadores como o ID, o *title* e a sua referência. Para identificar esse botão num *script*, é preciso considerar todos os identificadores para garantir que só existe um único conjunto com estes valores na página *web*. Ao considerar que para um único botão são necessárias várias informações para especificá-lo, a implementação de uma tarefa que envolva vários botões levará mais tempo. Para além disso, o processo de desenvolvimento de uma tarefa passa por várias fases como o planeamento, o desenvolvimento, testes e repetição em caso de erro de implementação. Estas fases podem repetir-se mais que uma vez e podem consumir mais tempo que o esperado.

Outro aspeto que se deve considerar, para além do tempo consumido no processo de implementação, é a garantia de previsibilidade. No ambiente de ciber segurança existem processos que requerem respostas no mínimo tempo possível e a qualquer momento do dia. Porém, é impossível garantir total fiabilidade na concretização destes processos visto que os colaboradores, sendo seres humanos, necessitam de descansar para conseguirem continuar o seu trabalho no seu próximo turno.

Para retirar a realização destas tarefas repetitivas dos trabalhadores, garantindo a previsibilidade e sustentabilidade de recursos humanos, foram criados conceitos e desenvolvidas novas ferramentas de automação.

Um desses conceitos é o *Robotic Process Automation* (RPA) que consiste no desenvolvimento de robots que executam as tarefas nas mesmas condições que o ser humano. Ou seja, interagem com as aplicações da mesma forma que o Homem. O outro conceito na área da automação centrado na gestão de tarefas na área de ciber segurança é *Security Orchestration Automation and Response* (SOAR). As tecnologias que se baseiam nestes conceitos podem executar processos que gerem milhões de dados

[1] Um programa com um conjunto de instruções que são executadas por outro programa

simultaneamente. As tecnologias podem melhorar a qualidade de trabalho do colaborador aumentando a sua disponibilidade para despender o seu tempo em tarefas que somente um ser humano é capaz de realizar, ou seja, que exijam um nível cognitivo superior e o poder de decisão.

1.2 Objetivos

Este projeto teve por objetivo principal desenvolver robots RPA para três casos de uso expostos pela equipa *Security Operation Center* (SOC) da MEO, nomeadamente:

- *Daily Reports from ArcSight to Tableau*;
- *Notify Customers of DoS Attacks with Reports*;
- *Notify Customers of DoS Attacks with Screenshots*.

Para além da implementação dos robots, foram feitas contribuições tais como:

- Foi feita uma análise das ferramentas utilizadas e testadas ao longo do projeto, *i.e.*, após a aprendizagem do funcionamento de cada uma delas e dos respetivos conceitos, foi realizada uma comparação final;
- Após o desenvolvimento dos robots foi feita uma avaliação final, através de métricas, que pretende avaliar os ganhos dos colaboradores (analistas SOC) com a colocação dos robots no ambiente de produção.

1.3 Organização do relatório

Os capítulos seguintes estão organizados da seguinte forma:

- O Capítulo 2 começa por fundamentar as ferramentas e intervenientes que participaram no projeto. Também se apresenta qual o problema que levou a criação das ferramentas de RPA e SOAR. Por fim, são explicadas o que são estas tecnologias e qual tem sido o seu impacto no mercado.
- O Capítulo 3 divide-se em três partes: a primeira foca-se nas ferramentas RPA e como se encontram no mercado atualmente, a segunda parte centra-se nas ferramentas usadas ao longo do projeto (*e.g.*, funcionalidades, componentes, nomenclaturas, entre outros), e a terceira parte apresenta uma tabela que resume as ferramentas *Automation Anywhere Enterprise*[2] e *Blue Prism*[3], comparando-as.
- O Capítulo 4 apresenta casos de uso pertencentes à equipa SOC e demonstra através de *System Sequence Diagram* (SSD) [2] como estes são executados pelos analistas. É feita uma análise dos mesmos para saber qual a tecnologia mais adequada. No fim da exposição de cada caso de uso, é apresentada uma proposta de implementação.
- No Capítulo 5 são apresentadas as implementações de cada caso de uso e em cada uma delas, quais as dificuldades que foram ultrapassadas.
- O Capítulo 6 apresenta os resultados dos robots implementados após a sua inserção no ambiente de produção. Os resultados são obtidos através de uma avaliação por métricas onde se compara a execução pelo robot e o analista SOC.

[2] https://www.automationanywhere.com/
[3] https://www.blueprism.com/

- O Capítulo 7 contém um sumário do trabalho realizado ao longo do projeto, juntamente com as conclusões principais e melhorias a concretizar.

Capítulo 2
Trabalho Relacionado

Este capítulo começa por fundamentar as plataformas *web* e as ferramentas do ambiente SOC usadas no projeto. Também são descritos os ataques decorrentes nos casos de uso. De seguida é feito um enquadramento do problema nos dias atuais e o que as tecnologias *Robotic Process Automation* (RPA) e *Security Orchestration Automation and Response* (SOAR) oferecem para solucionar.

2.1 Fundamentos

Com a evolução de tecnologia, a quantidade de informação a gerir aumentou. Este crescimento proporcionou ao aumento do número de processos a gerir e ao tratamento de eventos que ocorrem na rede das empresas.

Todas as empresas que possuem redes de média ou larga escala precisam um mecanismo de supervisão. Este mecanismo refere-se a equipas de *Security Operation Center* (SOC) que se define como uma organização que identifica, gere e mitiga os ataques que ocorrem na rede [9]. Equipas como o SOC são compostas por vários elementos chamados analistas.

Quando o número de máquinas da rede é muito superior ao número de supervisores, é preciso recorrer a mecanismos de monitorização. Estes mecanismos são ferramentas que são centradas na ciber segurança chamadas *Security Information and Event Management* (SIEM). Os SIEMs usados na MEO são *Micro Focus ArcSight*[4] e *AlienVault*[5].

O SIEM é um *software* que fica responsável por notificar os analistas dos eventos[6] que ocorrem na rede em tempo real [11] mas, para os notificar, é necessário que sejam criadas regras para saber os eventos que precisa endereçar.

Deste modo, os analistas implementam um conjunto de regras que definem o comportamento que pode corresponder a um ataque. Através destas regras, o SIEM monitoriza cada evento na rede e sempre que houver uma correspondência, são agregadas as informações desse evento e o analista é notificado do evento como um alerta. A Figura 2.1 descreve o processo de notificação do SIEM ao analista.

Entre o processo de deteção e notificação, o SIEM interage com outro *software* chamado *Request Tracker for Incident Response* (RTIR). Por norma, estas ferramentas servem para abrir uma estrutura *ticket* que faz a descrição de todos os componentes do alerta [15]. O *ticket*, quando criado, guarda todas as atualizações relacionadas com o alerta, sejam elas de intervenção humana

[4] https://www.microfocus.com/en-us/products/siem-security-information-event-management/overview
[5] https://www.alienvault.com/
[6] Um evento é um comportamento na rede como por exemplo, um *login* de um utilizador

ou não. Os sistemas de *ticketing* presentes no projeto são o RTIR e o *HP Service Manager Software* (HPSM)[7].

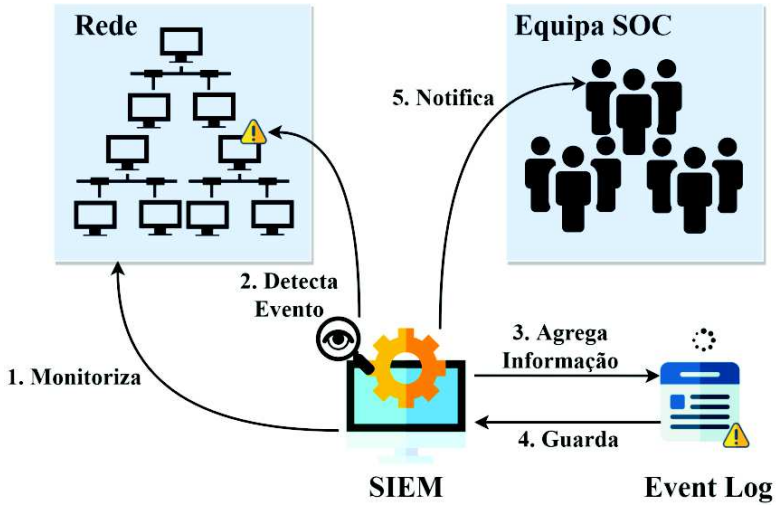


Figura 2.1: Processo desde da deteção de um suspeito ataque na rede (através do SIEM) até o analista ser notificado

Quando os analistas recebem a notificação do SIEM, têm a responsabilidade de averiguar o evento ocorrente. Acontece que nem todos os eventos são verdadeiramente ataques e para diferenciar um verdadeiro ataque de um falso, é usada a seguinte terminologia:

- **Verdadeiros Positivos** [12]: são alertas gerados pelos SIEMs que são ataques. Nesta situação o analista tem a responsabilidade de tratar deste alerta e mitigar o ataque que decorre;
- **Falsos Positivos** [13]: são alertas gerados pelos SIEMs que não são maliciosos. Estes alertas somente aumentam o "ruído" na rede e consomem tempo ao analista para a sua análise e tratamento.

Apesar dos SIEMs conseguirem reconhecer um evento com comportamentos estranhos, não conseguem parametrizar os comportamentos para definir qual o tipo ataque. Na ciber segurança existem vários tipos de ataques e todos os dias são exploradas vulnerabilidades na rede que incentivam a criação de novos ataques.

O ataque considerado neste projeto é *Denial of Service* (*DoS*) (negação de serviço). De acordo com [14], um ataque de *DoS* é considerado um dos ataques mais comuns na ciber segurança e caracteriza-se por um atacante impedir que os utilizadores legítimos das máquinas usufruam dos seus recursos. Uma das formas mais conhecidas é sobrecarregar a rede reduzindo a largura de banda ao utilizador. Este tipo de ataque torna-se mais sofisticado quando é executado a partir de vários sistemas, distribuídos, o que se denomina por *Distributed Denial of Service* (DDoS) (negação de serviço distribuída).

[7] https://www.xmatters.com/integration/hp-service-manager/

2.2 *Robotic Process Automation* (RPA)

De acordo com David Moss[8] referenciado em [20], o avanço das tecnologias levou as empresas a um *bottleneck* de tarefas sem se aperceberem disso.

Uma empresa é composta por vários departamentos que todos juntos resultam no bom funcionamento da mesma. Contudo, existem vários processos que para terminarem têm que passar por vários departamentos pois cada um deles é responsável por tarefas mais pequenas e importantes para cumprir o objetivo final do processo.

A Figura 2.2 ilustra como é possível ter uma ideia de como uma empresa funciona na concretização de um processo.

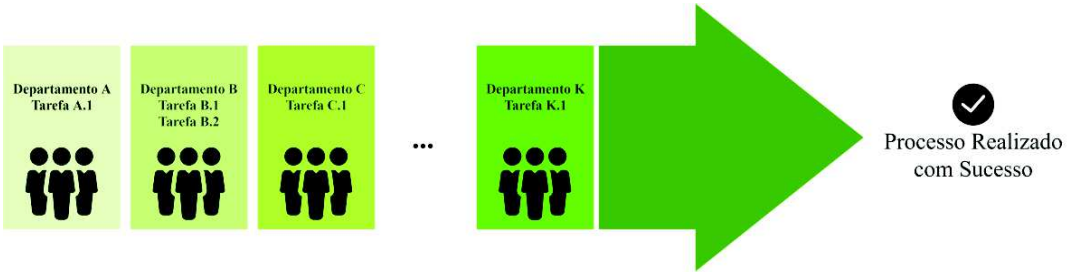


Figura 2.2: Exemplo da realização de um processo no ambiente empresarial

Tal como se apresenta na Figura 2.2, a progressão de um processo que envolva vários departamentos, depende que todas as tarefas anteriores sejam realizadas. Colaboradores que estão responsáveis por tarefas deste tipo de processos, realizam-nas várias vezes por dia e torna o exercício repetitivo e sem substância (*i.e.*, não usam as suas capacidades cognitivas para a sua execução). A repetição nas tarefas traz consequências como o cansaço, a falta de entusiasmo e a sensação de desvalorização do conhecimento e capacidade cognitiva do trabalhador.

Contudo, há colaboradores que, para além de serem responsáveis por tarefas como estas, têm outras de maior importância e que necessitam da intervenção humana para a sua realização. Como as tarefas repetitivas consomem mais tempo, diminuem a produtividade no trabalho.

A produtividade é um dos maiores objetivos de qualquer empresa. Esta, define-se pela regra de num dado *input*, se quer retirar o máximo de output num determinado espaço de tempo [20]. Numa empresa, processos como da Figura 2.2, são comuns e podem existir vários com a mesma estrutura para finalidades diferentes. Se for necessário realizar vários processos deste tipo, num curto espaço de tempo e com um número limitado de colaboradores, pode dar-se a um *bottleneck* provocando uma baixa na produtividade. A fórmula de produtividade apresenta-se na Figura 2.3.

$$\text{Productivity} = \frac{\text{Units of output}}{\text{Units of input}}$$

Figura 2.3: Fórmula de produtividade. Extraído de [22]

[8] *Chief Technology Officer* (CTO) da *Blue Prism*

O conceito *Robotic Process Automation* (RPA) surgiu para auxiliar os colaboradores na gestão de processos e baseia-se num *software* de automação, baseado em regras, que se aplicam em processos empresariais como em [21] e [24]. O RPA pode ser aplicado em processos de áreas distintas numa empresa, mas estes processos têm que ser estudados com cuidado para saber se de facto são ideais para a automação.

De acordo com o autor de [23], há critérios que se devem considerar para saber se um processo é viável para a automação:

- **Grande volume de transações**: um processo que envolva um número elevado de transações é adequado para o RPA visto que transações de grandes volumes têm tendência a ter processos repetitivos;
- **Acesso frequente a vários sistemas**: a realização de alguns processos envolve vários sistemas. Manualmente este tipo de trabalho é mais cansativo para o colaborador e por esse motivo torna necessária a automação;
- **Estabilidade do ambiente**: tem que se ter a certeza de que quando o processo é realizado, o ambiente é estável. Isto porque se não houver estabilidade, podem surgir resultados inesperados;
- **Intervenção limitada do Homem**: processos para automação têm que ter o mínimo de intervenção humana;
- **O mínimo levantamento de exceções possível**: quando um processo é automatizado, o número de exceções levantadas deve ser mínimo. Por norma, as exceções a levantar devem ser da própria tecnologia e não do processo em si. Contudo, caso seja do processo, deve-se encontrar uma alternativa para lidar com a situação inesperada;
- **Processos que levam ao erro**: devem ser automatizados processos que manualmente levam ao erro, com alto grau de probabilidade.

Sendo que há vários processos repetitivos dentro do contexto empresarial, o objetivo do RPA é conseguir retirar aos colaboradores essa repetibilidade e dar-lhes mais disponibilidade para tarefas que necessitam da sua intervenção para tomadas de decisão. Oferecer ao trabalhador a disponibilidade para concretizar tarefas de maior importância devolve-lhe o estímulo na realização do seu exercício e por isso há um aumento da produtividade. A Figura 2.4 ilustra como pode ficar a realização de um processo após a automatização.

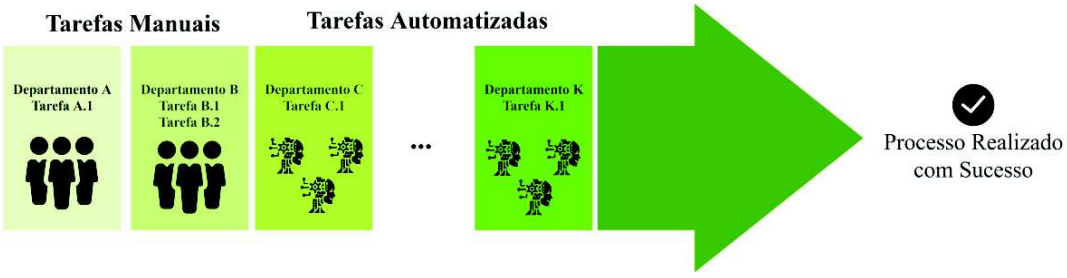


Figura 2.4: Ilustração da realização de um processo após a inserção de robots

Seguindo o raciocínio da Figura 2.4, mais de metade dos atores foram substituídos por robots de RPA. Apesar dos colaboradores terem sido substituídos, estes passam a fazer tarefas de maior relevância enquanto os robots fazem as tarefas repetitivas.

Inicialmente, o RPA foi um conceito muito discutido e que levantou algumas dúvidas ([28]) porque não tinham certezas de que um conceito simples com base em *workflows* seria capaz de

interagir com vários sistemas aumentando o *output*, a produtividade e o *Return of Investement* (ROI)[9].

Com o passar dos anos, o RPA tem evoluído e o mercado expandiu mais depressa que o esperado. Por exemplo, em [26] é apresentado um investimento de $300M por parte do *SoftBank* no RPA e o *UiPath* a partir de um investimento de $8M conseguiu recuperar $200M [27]. De acordo com a *Gartner*[10], o mercado de RPA no último ano cresceu cerca de 63% e o valor do RPA no mercado vale cerca de $846M [30]. A Figura 2.5 apresenta uma tabela que resume o crescimento das ferramentas de RPA no mercado no ano 2018.

2017 Rank	2018 Rank	Company	2017 Revenue	2018 Revenue	2017-2018 Growth (%)	2018 Market Share (%)
5	1	UiPath	15.7	114.8	629.5	13.6
1	2	Automation Anywhere	74.0	108.4	46.5	12.8
3	3	Blue Prism	34.6	71.0	105.0	8.4
2	4	NICE	36.0	61.5	70.6	7.3
4	5	Pegasystems	28.9	41.0	41.9	4.8
8	6	Kofax	10.4	37.0	256.6	4.4
11	7	NTT-AT	4.9	28.5	480.9	3.4
6	8	EdgeVerve Systems	15.7	20.5	30.1	2.4
7	9	OpenConnect	15.2	16.0	5.3	1.9
9	10	HelpSystems	10.2	13.7	34.3	1.6
		Others	273.0	333.8	22.2	39.4
		Total	**518.8**	**846.2**	**63.1**	**100.0**

Figura 2.5: *Software* de RPA no mercado (Valores apresentados em milhões de dólares). Imagem extraída de [31]

Neste momento, as previsões para o mercado do RPA continuam a ser positivas. Crê-se que o mercado continuará a crescer. De acordo com [32], prevê-se que no final de 2019 as três maiores empresas de RPA tenham um valor de $10.8B como ilustra a Figura 2.6.

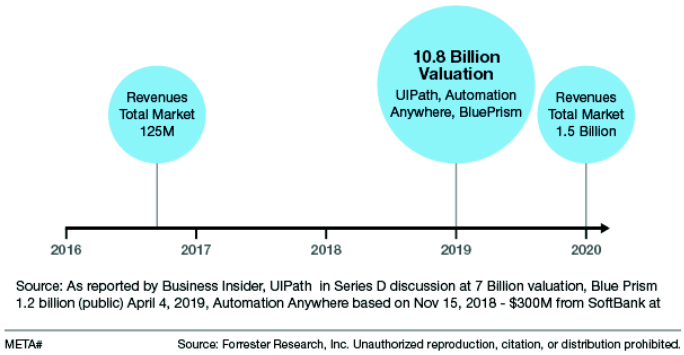


Figura 2.6: Previsão da *Forrester Wave* para 2019 e 2020. Imagem extraída de [32]

[9] É a medida para avaliar a eficiência de um investimento ou a comparação de um número de investimentos [29]

[10] https://www.gartner.com/en

2.3 *Security Orchestration Automation and Response* (SOAR)

Empresas como a MEO possuem redes de grande escala que as equipas SOC estão responsáveis por monitorizar. A tecnologia para monitorizar as redes mais conhecida e eficaz é o SIEM.

Acontece que em todos os segundos existem vários eventos a ocorrer e o SIEM regista-os todos, comparando-os às regras implementadas. Como é difícil modelar regras de modo a cobrir todos os eventos (representantes do tráfego), alguns deles são notificados aos analistas SOC e são falsos positivos. Outro problema reside no número escasso de elementos na equipa SOC, porque os colaboradores que têm conhecimento a nível de ciber segurança é limitado. Sendo que não há pessoas suficientes para avaliar e tratar todos os eventos rápida e eficazmente, alguns deles podem ser adiados (*i.e.*, são tratados no dia seguinte ou nas próximas horas).

Contudo, entre os falsos positivos que retiram tempo ao analista, para analisar e fechar o *ticket* respetivo, são adiadas outras tarefas de maior importância e que podem ser verdadeiros positivos. Para contribuir para a solução deste problema foi criada a tecnologia *Security Orchestration Automation and Response* (SOAR).

De acordo com a *Gartner* [34], o SOAR refere-se a tecnologias que permitem às organizações colecionar eventos monitorizados pelos SIEMs. O SOAR faz a análise dos eventos e através de um *workflow*, mitiga-os. Todas as etapas no SOAR são realizadas através de regras implementadas pela equipa SOC. Esta tecnologia pode-se considerar uma analogia ao RPA, mas esta concentra-se na área de ciber segurança. As vantagens da inserção desta tecnologia no SOC é o aumento da eficiência e produtividade nas operações, como também aliviar os analistas de falsos positivos permitindo-lhes tratar de eventos de maior relevância. O *workflow* do funcionamento do SOAR com as outras aplicações ilustra-se na Figura 2.7.

Figura 2.7: *Workflow* do SOAR

As características principais a considerar (de acordo com [33]) no SOAR são:

- **Integração de tecnologias**: o SOAR deve estar ligado às tecnologias (SIEMs e sistema de *ticketing*) para receber os eventos;

- **Automação de processos**: as etapas como a análise e mitigação são automatizadas através de regras criadas pelos analistas;
- **Gestão de incidentes usando uma abordagem *end-to-end***: desde a atribuição de propriedades, o registo dos detalhes do incidente até à tomada de decisão em conformidade às políticas da empresa;
- **Visualização dos dados**: relaciona-se com a documentação, os relatórios dos analistas e as métricas inseridas na ferramenta.

De acordo com [36], uma empresa chamada *Swimlane*[11] que vende um produto de SOAR apresentou um caso de um cliente ter sido vítima de 200 ataques de *phising*[12] no espaço de uma semana. A equipa SOC levou quatro horas para tratar cada alerta, mas após a instalação do SOAR e criação das regras, cada alerta foi tratado em 15 minutos.

Exemplos como do *Swimlane* garantem boas previsões para o SOAR no mercado. De acordo com [38], a *Gartner* prevê, em 2021, que o mercado do SOAR vai expandir e 70% das empresas que se dediquem ao SOC virão a incluir o SOAR no ambiente de ciber segurança.

2.4 Aplicações das tecnologias RPA em Ciber Segurança

Como mencionado ao longo do capítulo, a ciber segurança é uma área que está em constante evolução. Com o aparecimento de novos ataques, a quantidade de informação cresce.

Como mencionado em [40], em 2011 já era feita uma análise ao conjunto de controlos de segurança baseados na Norma ISO 27001[13] que podiam ser automatizados para a melhoria da eficiência dos processos de gestão da informação. Na Norma existem 133 controlos e 11 domínios, cada controlo pertence a um domínio e de acordo com a análise feita, somente 37 controlos têm a possibilidade de serem automatizados.

De momento, os SOCs têm os processos parcialmente automatizados, ou seja, através dos SIEMs, os comportamentos estranhos são detetados, os analistas são notificados e responsabilizam-se por filtrar os falsos positivos e mitigar os restantes. Contudo, com redes de grande escala, são detetados milhares de eventos ao mesmo tempo e não há analistas suficientes para tratar de todos rapidamente. Por esse motivo, em [41] (uma opinião elaborada por Kumar Saurabh[14]), o RPA deveria ser inserido no SOC para atingir os objetivos básicos para o seu bom funcionamento.

Em [42] referencia-se que existem inúmeros desafios na ciber segurança que podem ser ultrapassados com o auxílio do RPA, tais como:

- Os dados gerados numa empresa crescem de forma abundante e são enviados para os clientes, porém a comunicação entre a empresa e os clientes é ampla. Devido a esta amplitude, podem surgir vários pontos que levem à tentativa de ataque;
- Muitas das tarefas realizadas são repetitivas que não exigem poder de decisão;

[11] https://swimlane.com/

[12] Ataque que incentiva ao utilizador a revelar os dados pessoais. A forma mais comum é através de *e-mails* falsos ou *websites* [37]

[13] https://www.27001.pt/

[14] *Chief Executive Officer* da ferramenta *LogicHub* (https://www.logichub.com/, tecnologia SOAR)

- Entre outras.

Nas referências [43] e [44], os objetivos principais para a inserção do RPA em ciber segurança são: com a automatização de processos, usufruir do número limitado de profissionais que tenha conhecimentos relativos à segurança informática com outros processos mais relevantes; redução do tempo de execução dos processos que detetam ameaças de ataques; diminuição do número de humanos a interagir com informações pessoalmente identificáveis (PII[15]).

Em [21], os autores recrearam um processo de negócio inserindo, parcialmente, a tecnologia RPA no mesmo. O tempo de observação foi de uma semana e o maior benefício encontrado foi o aumento da produtividade dos colaboradores que ainda participavam no processo. Em termos de comparação, o processo com a tecnologia RPA conseguiu tratar mais 21% dos casos face ao processo totalmente manual. Contudo, não se observou grande diferença no tempo de execução do processo pois, em média, houve uma redução de nove segundos face ao tempo de execução realizado somente por humanos. Os ganhos principais, neste estudo, foram o aumento da produtividade e a diminuição dos custos (de acordo com o estudo).

Os autores de [47] estudaram os ganhos obtidos através do RPA na empresa Telefónica O2. Este estudo foi feito em Abril de 2015 onde foram automatizados 160 subprocessos entre os quais pertencem a 15 processos principais de negócio da empresa. O número de transações mensais passou a ser entre 400 000 a 500 000, houve um ROI entre os 650 e os 800 por cento e o tempo total para receber o valor do investimento foi de 12 meses.

Em [45] são apresentados vários processos de negócio de diferentes áreas que já foram implementados com a tecnologia RPA. Contudo, não foram ainda implementados quaisquer processos de negócio que pertençam à área de ciber segurança.

Em [48], Brocke *et al.* fizeram o estudo sobre quais as expectativas sobre os sistemas tecnológicos. Os autores consideram quatro tipos de sistemas de trabalho tal como se apresenta na Figura 2.8.

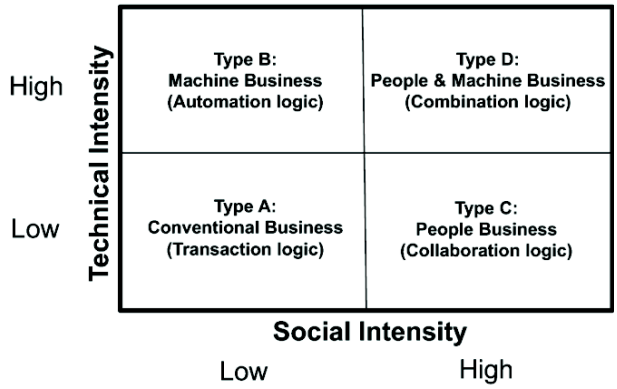


Figura 2.8: Os quatro tipos de sistemas de trabalho futuros. Extraído de [48]

[15] *Personally Identifiable Information*

O tipo B (*Type B*) foca-se em sistemas que tenham alto conhecimento técnico mas baixa interação social. Este tipo de sistemas são representados por áreas de automatização como o RPA, a Inteligência Artificial e *Machine Learning*[16].

O tipo B associa-se a *Intelligent Enterprise Systems*[17] (IES) que foi descrito pelo autor Alexander Maedche. O autor acredita que, a nível económico, o IES pode aumentar a produtividade ao automatizar determinadas tarefas e pode também aumentar o desempenho. Para além dos ganhos económicos, Alexander acredita que é possível atingir ganhos sociais como, por exemplo, os colaboradores conseguirem contribuir mais na empresa através da realização de processos mais complexos e podem também encontrar equilíbrio entre o trabalho e a sua vida pessoal.

[16] *Machine Learning* é uma subárea da Inteligência Artificial que providencia aos sistemas a habilidade de aprender e melhorar o comportamento através da experiência sem estar expressamente programado [49]

[17] *Intelligent Enterprise Systems* é a junção de *Enterprise Systems* (por exemplo, *Customer Relationship Management*) com *Information Systems*.

Capítulo 3
Ferramentas

Este capítulo apresenta as ferramentas que foram utilizadas durante o projeto. A Secção 3.2 é dedicada ao *Automation Anywhere Enterprise* (AAE) que é um dos *softwares* de RPA mais vendidos no mundo. Para além do AAE, foi usado o *Blue Prism* (BP) que se descreve na Secção 3.3 – outra ferramenta de RPA sendo uma das maiores concorrentes do AAE. A nível de SOAR, apresenta-se o *ATAR Labs* na Secção 3.4.

3.1 Ferramentas de RPA

Atualmente o mercado de RPA está em expansão e já existem inúmeras ferramentas desta tecnologia. De acordo com a *Forrester Wave*[18], há três ferramentas líderes: o *UiPath*[19], o *Automation Anywhere* e o *Blue Prism*. A Figura 3.1 representa o quadrante mágico das ferramentas de RPA em 2018.

Figura 3.1: Quadrante Mágico de RPA do ano 2018. Imagem extraída de [4]

[18] https://go.forrester.com/
[19] https://www.uipath.com/

Apesar de todas as ferramentas terem a finalidade de automatizar processos, possuem características diferentes. A avaliação feita pela *Forrester Wave* dependeu de três métricas:

- *Current Offering*: corresponde ao eixo vertical da Figura 3.1 que indica qual o peso da oferta da ferramenta atualmente;
- *Strategy*: corresponde ao eixo horizontal da Figura 3.1 e indica qual a força das estratégias dos vendedores das ferramentas;
- *Market presence*: corresponde ao tamanho dos círculos de cada ferramenta representados na Figura 3.1.

Estas métricas subdividem-se em vários componentes onde cada um deles possui determinado peso percentual ([4]). A avaliação utilizou a escala de zero a cinco onde o zero era considerado o valor mais fraco e cinco o valor mais forte. Os resultados da avaliação apresentam-se na Tabela 1:

Ferramenta	*Current offering*	*Strategy*	*Market Presence*
UiPath	4.25	4.70	5.00
Automation Anywhere	3.97	4.70	5.00
Blue Prism	3.63	4.50	5.00

Tabela 1: Resultados finais da avaliação das três ferramentas líderes de RPA

Para além das métricas usadas na avaliação, existem outras três que devem ser consideradas e revelam grande peso:

- Modo de aprendizagem: a forma de um utilizador aprender a usar uma ferramenta. Ou seja, se a ferramenta tem uma versão *Trial* para explorar;
- Curva de aprendizagem: o utilizador é capaz de compreender a lógica da ferramenta e construir um processo facilmente?
- Popularidade no *Google*: sendo este um dos motores de busca mais reconhecidos no mundo, qual a ferramenta mais popular.

A Figura 3.2 apresenta um resumo das três métricas sobre os líderes de RPA.

Features	UiPath	Blue Prism	Automation Anywhere
How will I learn?	Has a community edition/ Free edition available	No trial version available	Trial version is available for 30 days
Learning Curve	Has a user-friendly visual designer	Has a user-friendly visual designer, easier than Automation Anywhere	Developers friendly but requires high programming skills
Google Trends Popularity	Most Popular tool	More popular than Automation Anywhere	Least popular tool in the trio

Figura 3.2: Resumo das últimas três métricas sobre os líderes de RPA. Tabela extraída de [3]

Sendo assim, adicionando estas três métricas é possível concluir que a ferramenta *UiPath* é considerada (através das avaliações apresentadas), a melhor ferramenta de RPA no mercado atualmente. Devido à sua popularidade e dando a possibilidade a qualquer utilizador usufruir de uma versão grátis, garante o seu lugar no mercado. O *Automation Anywhere* fica um lugar abaixo devido à limitação de 30 dias para usufruir a ferramenta e requer que o utilizador tenha alguns conhecimentos a nível de programação. Por último, o *Blue Prism* apesar de ser mais popular que

o *Automation Anywhere*, não garante uma posição superior por não oferecer uma versão *Trial* aos utilizadores, e por esse motivo, tem menos reconhecimento.

No contexto deste projeto as ferramentas de RPA testadas foram *Automation Anywhere* e *Blue Prism*.

3.2 *Automation Anywhere Enterprise* (AAE)

O *Automation Anywhere* é uma das empresas norte-americanas que desenvolve *software* de RPA e dedica-se ao desenvolvimento deste *software* para automatizar processos de forma eficaz e sem precedentes nas empresas ([16]). O AAE só pode ser executado nos sistemas operativos Windows e não é possível executar os robots em todos os *browsers* como se pode ver na Tabela 2.

Browser	Executável?
Internet Explorer	É possível executar os robots
Google Chrome	Para ser possível executar os robots, é preciso instalar um *plugin* da ferramenta
Mozilla Firefox	Não é possível executar os robots

Tabela 2: Execução dos robots nos *browsers* na ferramenta *Automation Anywhere Enterprise*

Para além das restrições relacionadas com os *browsers*, o ambiente onde é feita a implementação e execução dos robots deve ter grande disponibilidade em termos de memória – *System Requirements* exige 16 GB de RAM.

3.2.1 Componentes do AAE

O AAE é dividido em duas componentes: o *Control Room* (assemelha-se a um servidor) e o cliente.

O *Control Room* é o componente central da ferramenta onde é possível registar os clientes e ver o histórico de ações realizadas pelos mesmos (*e.g.*, *login*, *logout*, *upload* de um robot, *download* de um robot, execuções). Para além disso, tem *dashboards* que apresentam graficamente o que foi executado enquanto a ferramenta foi usada – *e.g.*, os robots que foram executados com sucesso (ou não), quantas vezes executaram, a que horas iniciaram e quando terminaram. Também é possível agendar a execução dos robots.

Existem dois tipos de clientes: *BotCreator* e *BotRunner*. O *BotCreator* representa o cliente *developer* que implementa os robots com os comandos disponibilizados pelo AAE. Enquanto o *BotRunner* responsabiliza-se por agendar e executar os robots implementados. Para que os robots sejam executados, é necessário pelo menos um *BotRunner* conectado ao *Control Room*. As janelas das duas componentes podem ser vistas na Figura 3.3.

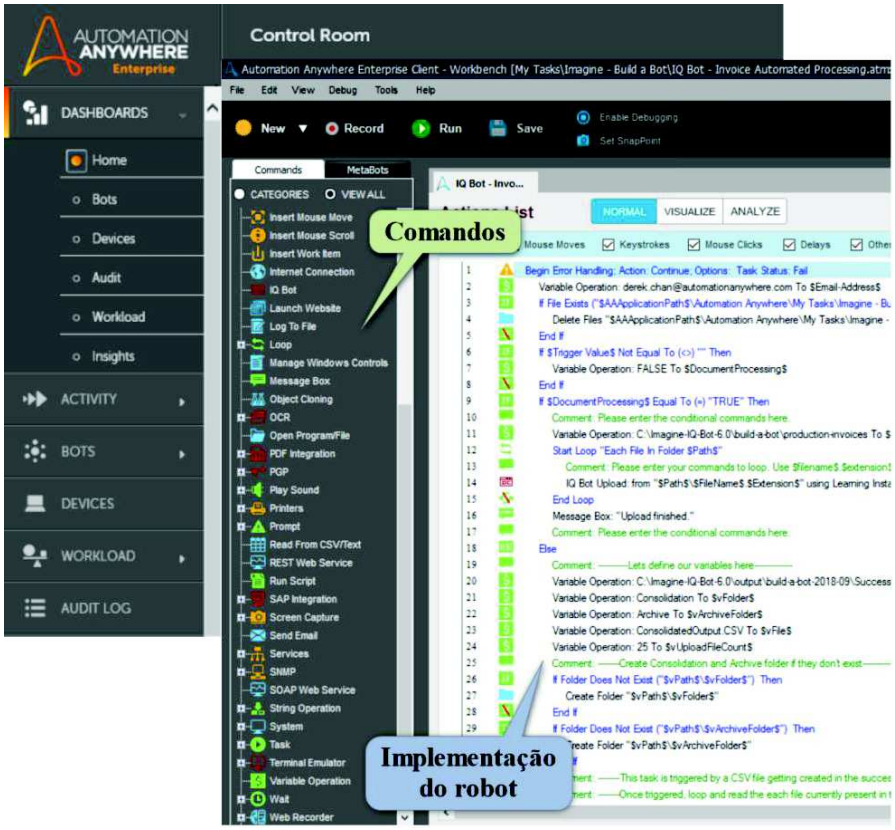


Figura 3.3: Janelas do *Control Room* e implementação. Imagens extraídas de [6] e [7]

Na Figura 3.4 são ilustradas as ligações entre os componentes. Ambos os clientes estão ligados ao *Control Room* e é necessária a criação de (pelo menos) um cliente de cada tipo para o bom funcionamento da ferramenta.

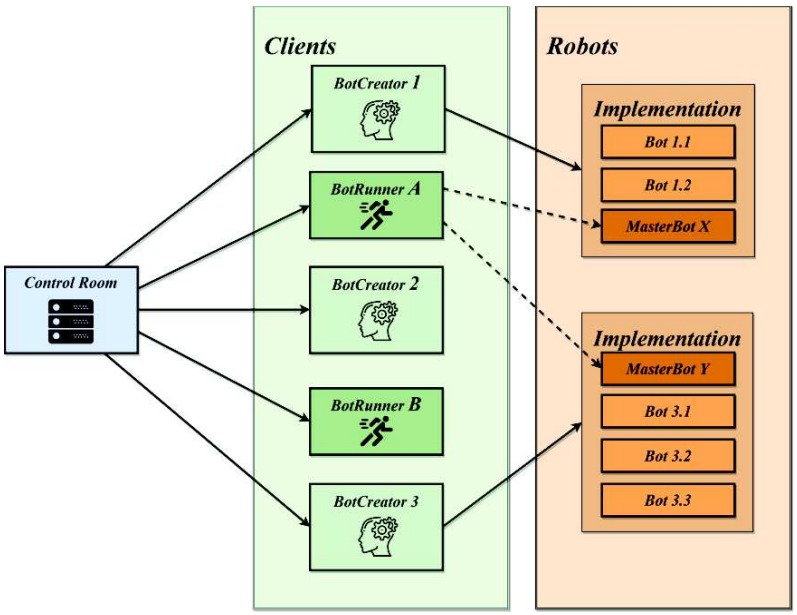


Figura 3.4: Esquema das conectividades entre os componentes: *Control Room* e clientes

Na Figura 3.4 vê-se que os clientes estão ligados diretamente ao *Control Room* porque são todos criados nesse componente. Os clientes *BotCreators* podem criar vários robots, tantos quantos quiserem. Para exemplificar, temos na Figura 3.4 o bloco *Robots* onde se situam todos os robots implementados. Os robots presentes no bloco, foram criados por *BotCreators*. Uns robots criados pelo *BotCreator 1*, outros pelo *BotCreator 3*. Apesar dos robots terem sido criados por um cliente específico, outros clientes podem aceder e fazer alterações. No caso dos *BotRunners*, temos o exemplo do *BotRunner A* que acede aos robots *MasterBot X* e *MasterBot Y* para os agendar no *Control Room*. Relativamente ao acesso dos *BotRunners* aos robots para agendamento é total.

3.2.2 Bots, Masterbots e Metabots

A terminologia do AAE inclui Bots, Masterbots e Metabots.

Um **Bot** é o conjunto de comandos implementados.

Um **Masterbot** é composto por vários Bots.

Deste modo evita-se que um Bot tenha um número elevado de comandos (aumentando a eficiência de execução do Masterbot) e promove-se a reutilização de Bots. Por exemplo, se forem criados três Bots para uma aplicação ou plataforma – um para *login*, um que faça a tarefa principal e, um para *logout* – os Bots inicial e final podem ser reutilizados se existir outro caso de uso na mesma aplicação. Este exemplo ilustra-se na Figura 3.5.

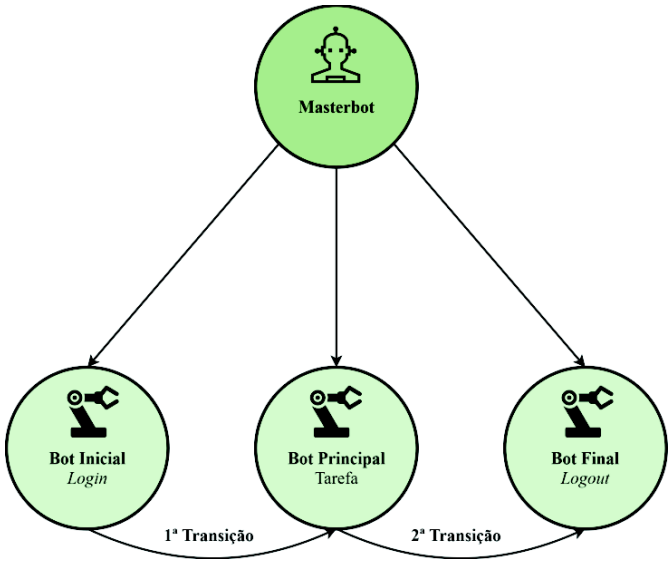


Figura 3.5: Exemplo de um Masterbot composto por três Bots

Os **Metabots** são por base como os Bots, mas têm maior abrangência nas funções a usar. Ou seja, para além das funcionalidades oferecidas pela ferramenta, podem incluir funções "extra" já criadas em ficheiros *Dynamic-link Library* (DLL). A Figura 3.6 ilustra o começo da implementação de um Metabot onde é necessário definir qual a aplicação em que o Metabot será empregue.

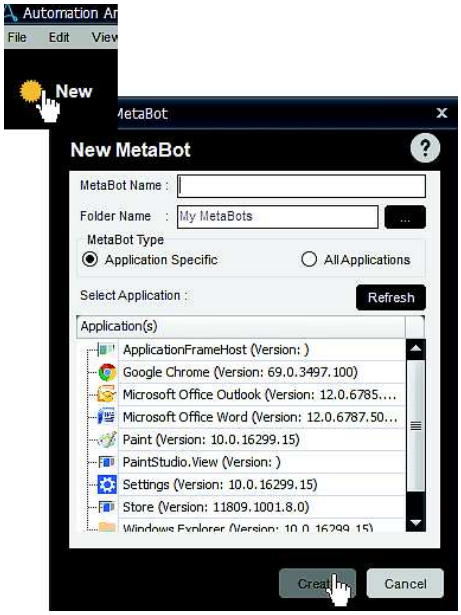


Figura 3.6: Janela inicial para implementar um Metabot

A Figura 3.7 apresenta um exemplo de uma janela de implementação onde é programada a lógica do Metabot. No caso dos Metabots, aparece ao lado direito uma janela que exibe todas as funções do ficheiro DLL disponíveis para o *BotCreator* utilizar[20].

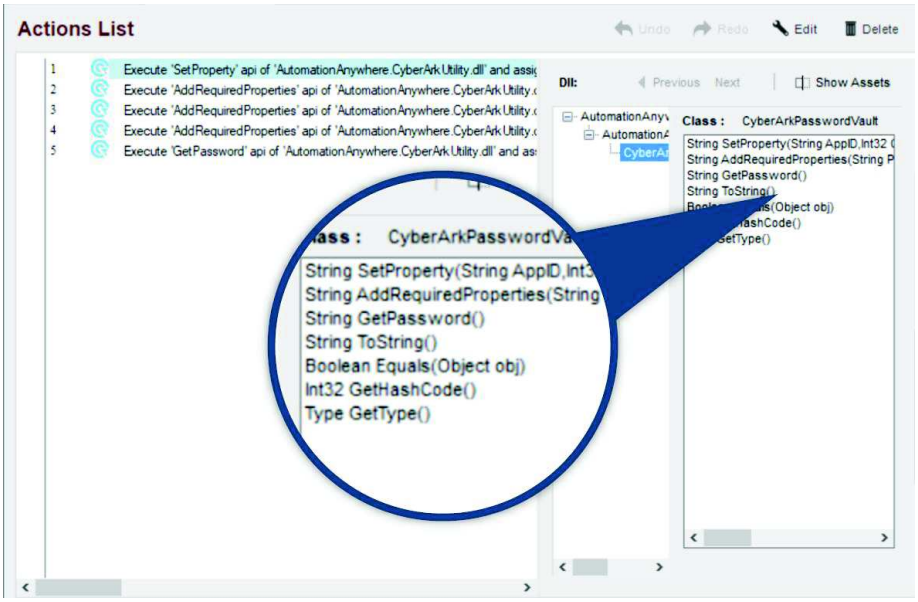


Figura 3.7: Janela de implementação de um Metabot. Imagem extraída do AAE

[20]Também é possível o *BotCreator* criar um ficheiro DLL com as funções que precisa e usá-lo no Metabot

3.2.3 Funcionalidade *Recorder*

Na implementação de um Bot cada passo corresponde a um comando. Contudo, existem passos que podem ser gravados pela ferramenta e geram automaticamente os seus respetivos comandos. A esta funcionalidade chamamos de ***Record***.

A funcionalidade *Record* grava os passos realizados pelo cliente e converte-os nos comandos correspondentes da ferramenta – *e.g.*, carregar numa série de botões de seguida. O cliente não precisa de tratar de cada *click* manualmente. Para este tipo de ocasiões o *Record* é o mais adequado. Se o cliente carregar num botão, o comando é gerado automaticamente e adiciona-o à *Task*, proporcionando rapidez na implementação do Bot. Esta ferramenta é dividida por três tipos de *Record*: *Smart Recorder*, *Screen Recorder* e *Web Recorder*.

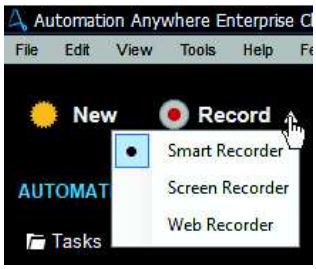


Figura 3.8: Funcionalidade *Record* e os tipos de gravação existentes

3.3 *Blue Prism* (BP)

O *Blue Prism* é uma empresa que desenvolve *software* para RPA cuja ferramenta tem o mesmo nome da empresa, tendo sido desenvolvida em Inglaterra. Esta empresa está a fazer desenvolvimentos, no mercado de RPA, na área de Inteligência Artificial (IA) como por exemplo, melhorar o reconhecimento de caracteres em ficheiros digitalizados em formato PDF (*Portable Document Format*). O *Blue Prism* só é executado somente em ambientes que tenham sistemas operativos Windows e de momento, consegue interagir com os *browsers Internet Explorer*, *Google Chrome* e *Mozilla Firefox*.

3.3.1 Componentes do *Blue Prism*

O *Blue Prism* é dividido por três componentes: o *Control Room*, o cliente e os *dashboards*.

O *Control Room* é o componente responsável pela gestão dos processos e é composto por processos, recursos e sessões, conforme ilustrado na Figura 3.9.

Os processos correspondem às implementações dos casos de uso. Para que os processos sejam vistos no *Control Room*, é necessário publicá-los após a sua implementação. Um recurso é a unidade – *i.e.*, máquina – responsável pela execução do processo. Sempre que se abre o *Control Room*, é possível ver todas as máquinas que têm conexão com a ferramenta – e quais estão *online* em tempo real. Para reconhecimento do estado da máquina, existe uma propriedade que indica o estado atual: *offline, connection lost* ou *owned by*, por exemplo. Uma sessão corresponde à associação de um processo a um recurso. Dentro de uma sessão é possível ter vários estados – *e.g. Pending*, *Running*, *Complete, Terminated.*

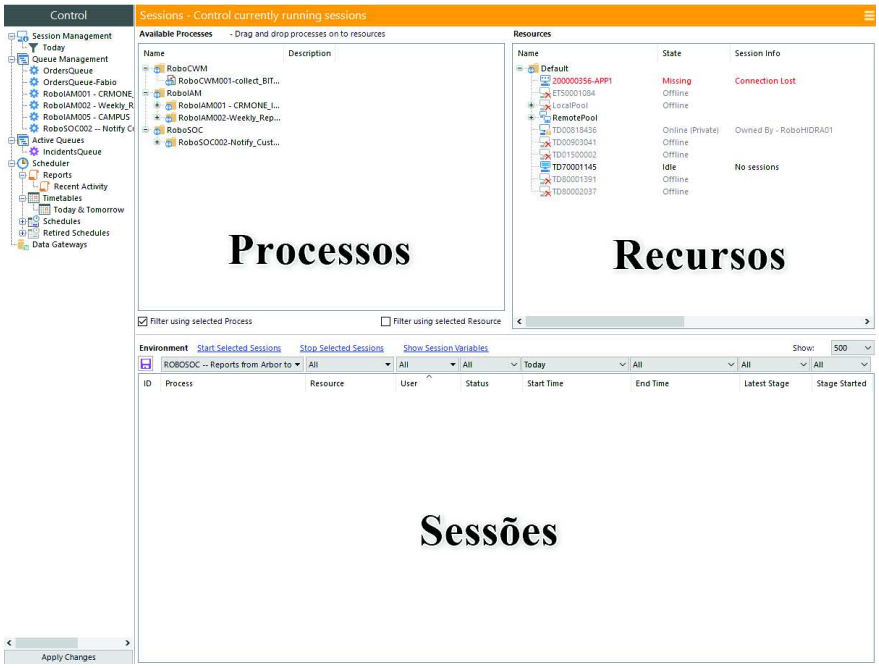


Figura 3.9: Página do *Control Room* no *BP*. Imagem extraída do *BP*

O cliente é a representação do humano que interage com a ferramenta. Podem ser criados vários clientes, tantos quanto necessários. Sempre que um cliente for criado escolhe-se o papel que terá na ferramenta. Existem vários papéis já criados na ferramenta, como por exemplo *Developers*, *Testers*, *RunTime Resources*, entre outros. As permissões dadas na ferramenta dependem do papel que foi atribuído ao cliente, e as funcionalidades para as quais não foram dadas permissões ficam bloqueadas. Uma particularidade dos papéis é a possibilidade de criar novos elementos deste tipo, *i.e.*, criar um perfil próprio.

Os *dashboards* fazem a apresentação gráfica, conforme Figura 3.10, de vários indicadores. Por exemplo, informações dos robots que foram criados e que estão a ser utilizados de momento, os que foram executados com sucesso – ou não – quantas vezes executaram, a que horas iniciaram os robots e quando terminaram.

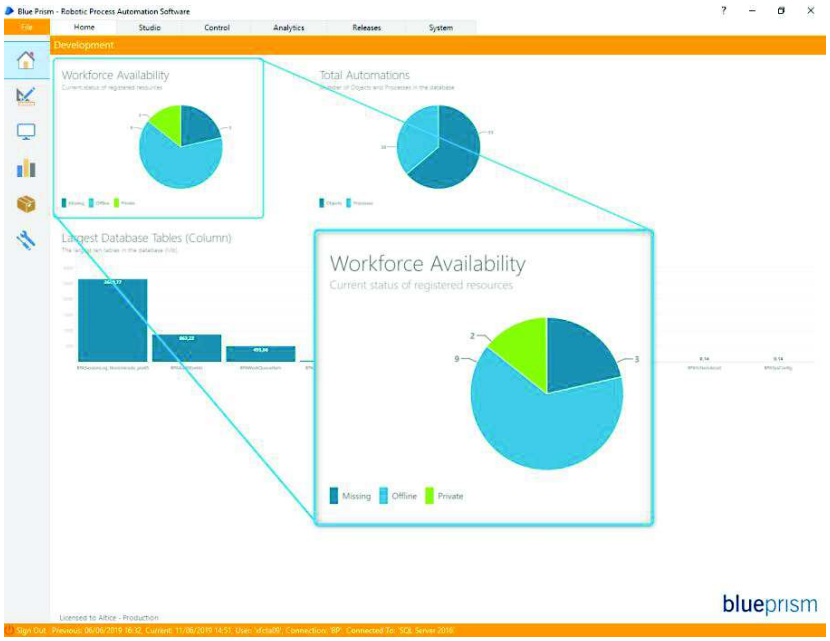


Figura 3.10: Página dos *dashboards* no *BP*. Imagem extraída do *BP*

3.3.2 *Object Studio* e *Process Studio*

A implementação dos robots no *Blue Prism* é realizada em dois ambientes: *Object Studio* e *Process Studio*. O *Object Studio* é o ambiente criado para realizar interações com as aplicações. O *Process Studio* não interage com as aplicações mas faz o processamento do caso de uso.

O *Object Studio* possui a componente *Application Modeller*, por forma a interagir com as aplicações (ilustrado na Figura 3.11). Cada *object* – processo no *Object Studio* – tem um *Application Modeller* e só pode estar associado a uma aplicação. As aplicações podem ser *browsers*, aplicações da Microsoft, aplicações Java, entre outras.

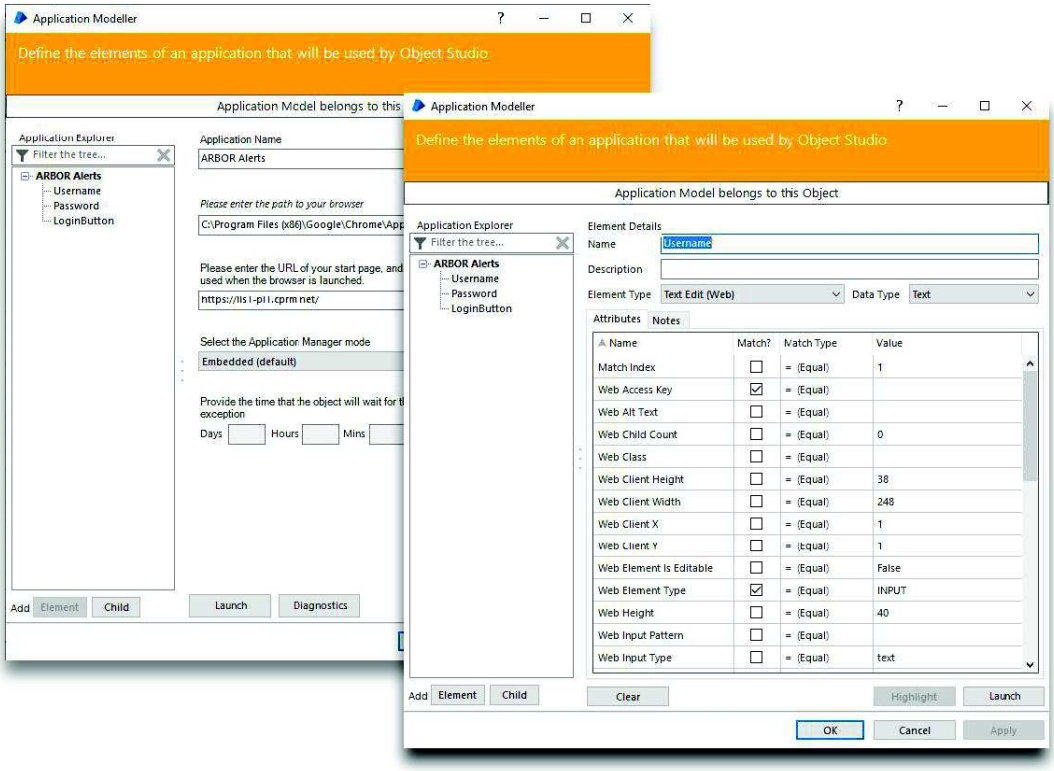


Figura 3.11: *Application Modeller* no *BP*. Atrás estão as propriedades da aplicação associada e à frente os elementos identificados. Imagens extraídas e editadas do *BP*

Quando uma aplicação é associada ao *Application Modeller*, esta contém elementos – como caixas de texto, *links*, imagens, botões, entre outros. O *Blue Prism* interage com estes elementos do mesmo modo que um humano, através do *Application Modeller* – que identifica e guarda cada elemento para a automatização do caso de uso. Para a identificação de elementos nas aplicações, é preciso fazer *Launch* da aplicação a partir do *Application Modeller.* Após a abertura da aplicação, é feita a operação *spy*. A operação *spy* é a identificação dos elementos na aplicação e existem quatro modos base: HTML *Mode*, *Win32 Mode*, *Region Mode* e *Acessibility Mode.* Para determinadas aplicações, existem modos específicos que o próprio *Blue Prism* ativa após a associação da aplicação.

Tal como está ilustrado na Figura 3.12, o *Object Studio* e no *Process Studio* distinguem-se somente em seis operações: *read*, *write*, *navigate*, *code*, *wait* e *alert*. Todas as operações no *Object Studio* estão relacionadas com interações de elementos obtidos através do *Application Modeller*.

Os propósitos das operações são:

- **Read**: faz a leitura de um elemento, *e.g.*, lê uma tabela existente numa página web;
- **Write**: faz a escrita sobre um elemento, *e.g.*, uma *tab* de pesquisa;
- **Navigate**: seleciona um elemento, carregar num elemento – com o botão esquerdo ou direito do *mouse* –, insere texto, faz *Launch* ou *Terminate* da aplicação, entre outras funções;
- **Code**: insere-se um *script* sobre um elemento ou mesmo na própria aplicação. Também pode-se inserir um *script* para o *object*, *e.g.*, inserir um *pop-up*;
- **Wait**: esta funcionalidade espera até o aparecimento de um elemento. Insere-se o tempo máximo – à nossa escolha – na função e resulta um de dois caminhos:
  - **Se carregar a tempo**: continua o processo;
  - **Se não carregar a tempo**: lança-se uma exceção.

O *Process Studio* tem a operação chamada **Alert** que apresenta uma mensagem de alerta, *i.e. pop-up*, quando o processo é usado no *Control Room*.

No *Process Studio* é feita a gestão dos dados do caso de uso (que podem ter sido obtidos no *Object Studio*, ou não). Para que um *object* seja usado dentro de um *process* – processo do *Process Studio* – tem que ser chamado através de uma operação **Action**.

Uma boa política de utilização do *Blue Prism* é implementar o mínimo de operações possíveis em cada *object* ou *process*. Isto serve para flexibilizar o processo, ou seja, quando uma alteração for realizada sobre o mesmo, os impactos dessa alteração sejam mínimos.

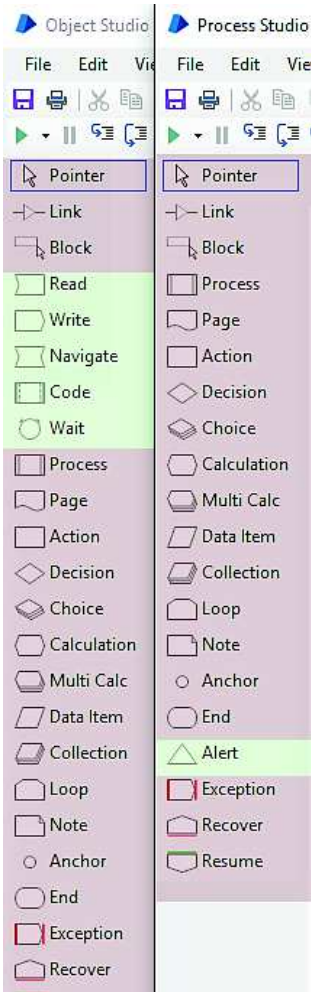


Figura 3.12:Funcionalidades no *Object Studio* e *Process Studio*. Imagens retiradas do *BP*

Visual Business Object (VBO)

Quando o *Blue Prism* é instalado, vem com as operações apresentadas na Figura 3.12. A operação **Action** executa pequenas funções que advêm da própria instalação. Um exemplo onde estas funções são usadas, é a *collection.* Uma *collection* é uma estrutura que pode conter vários dados ao mesmo tempo e de tipos diferentes. É uma estrutura mutável, tendo funções que interagem com a estrutura. Estas funções chamam-se de *Internal Business Objects* (IBO).

Para além destas funções "base", o *Blue Prism* tem na sua diretoria de ficheiros XML que contêm funções "extra". Ou seja, é possível fazer mais interações, sobre a mesma estrutura. Estas funções "extra" chamam-se *Visual Business Objects* (VBO). Na Figura 3.13 são ilustradas as funções base (IBO) e as funções "extra" (VBO) sobre uma *collection*.

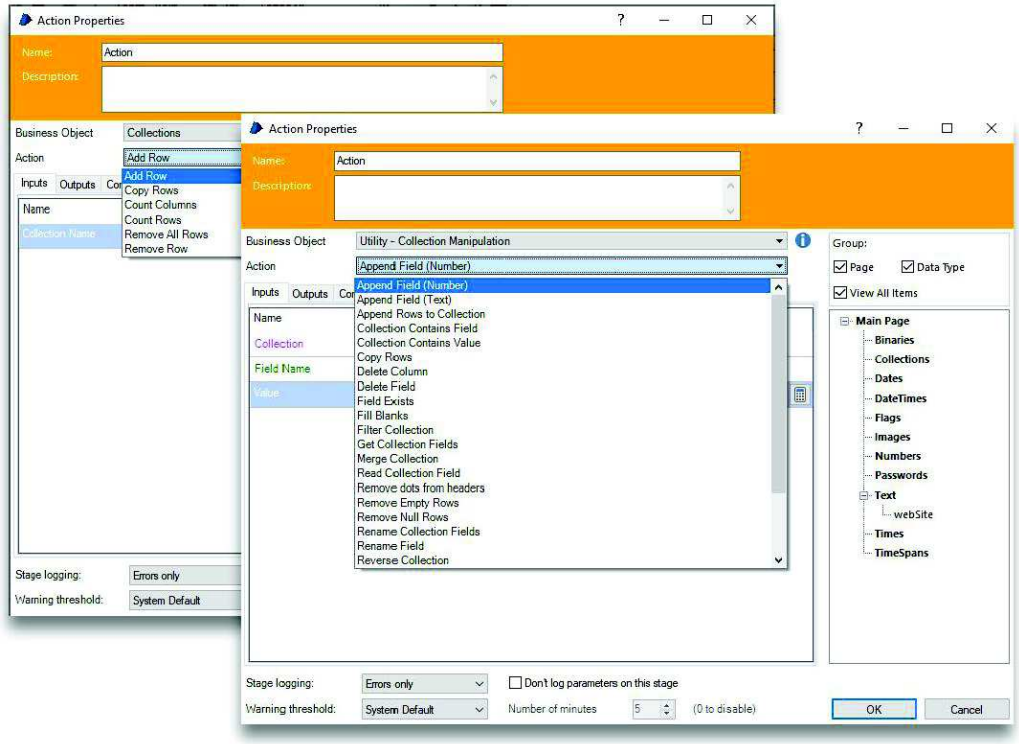


Figura 3.13: Funções base e funções "extra" importadas de um ficheiro XML. Imagens obtidas do *BP*

3.3.3 Implementação dos robots

A implementação no *Blue Prism* apresenta-se em forma de *flowcharts* – em português, diagramas de fluxo. A vantagem da exposição de processos em forma de *flowchart* visa proporcionar facilidade e compreensão na leitura. Para além disso, é possível ver o caminho que é feito durante a execução do processo. Um exemplo de um *flowchart* apresenta-se na Figura 3.14.

Na Figura 3.14 está representado um exemplo que faz a soma de duas variáveis *Number* e *Digit* e verifica se o resultado (guardado na variável *Result)* é superior a seis. Se for superior a seis, é dado como terminado o processo. Se for inferior ou igual, então é adicionado cinco ao *Result* e o processo é terminado.

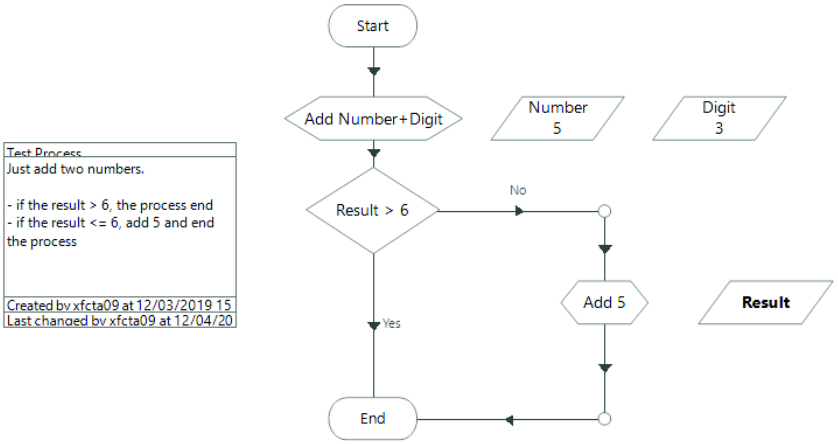


Figura 3.14: Processo exemplo no *Blue Prism*

Sempre que é criado um processo, existem dois estados obrigatórios: *Start* e *End*. O estado *Start* só pode existir um – isto porque sempre que o processo inicia, a máquina deve sempre estar no mesmo estado – mas o estado *End* pode existir várias vezes no mesmo processo. Isto porque um processo pode terminar com estados diferentes.

Quando um processo é executado em modo *debug*, é possível acompanhar em que estado estão as variáveis e em que passo está o processo. O fator que dá conhecimento ao cliente do ponto de situação do processo é a cor amarela sobre cada função. Com o exemplo apresentado na Figura 3.14 é ilustrada na Figura 3.15 como é apresentado o processo quando está em modo *debug*. Quando ocorre um erro na execução do processo neste modo, o processo pára automaticamente e é lançada uma mensagem de erro.

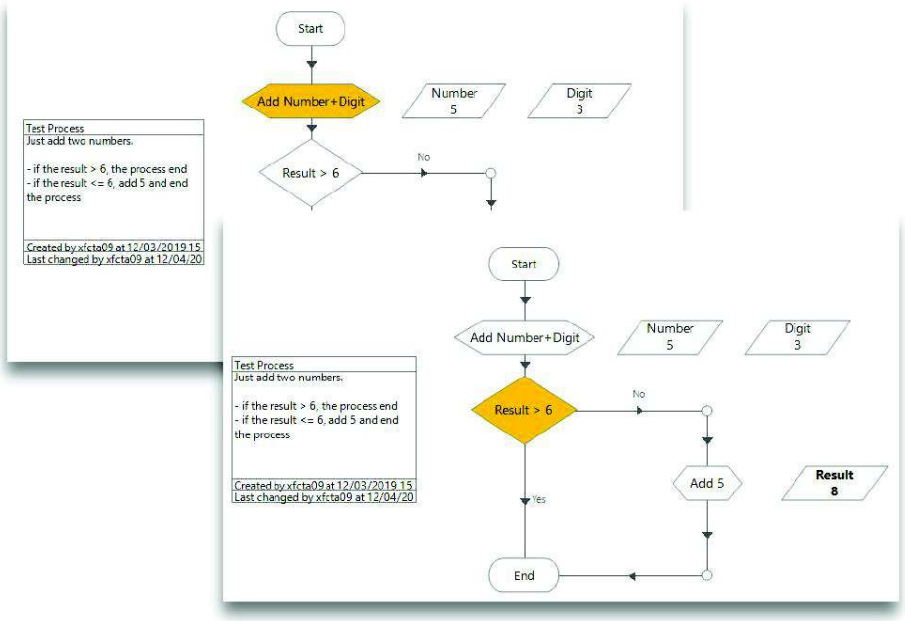


Figura 3.15: Demonstração da execução de um processo em modo *debug*

3.3.4 História de *objects* e *processes*

Quando um *object* – ou *process* – é implementado, o novo estado da sua implementação deve sempre ser guardado. O *Blue Prism* tem a funcionalidade de obrigar o cliente a descrever o que foi implementado, ou que alterações foram feitas, sempre que tenta guardar o processo. A esta funcionalidade chamamos de **História**.

A história descreve a evolução do robot e é possível ler todas as alterações feitas e descritas pelo cliente. Um exemplo de uma história ilustra-se na Figura 3.16.

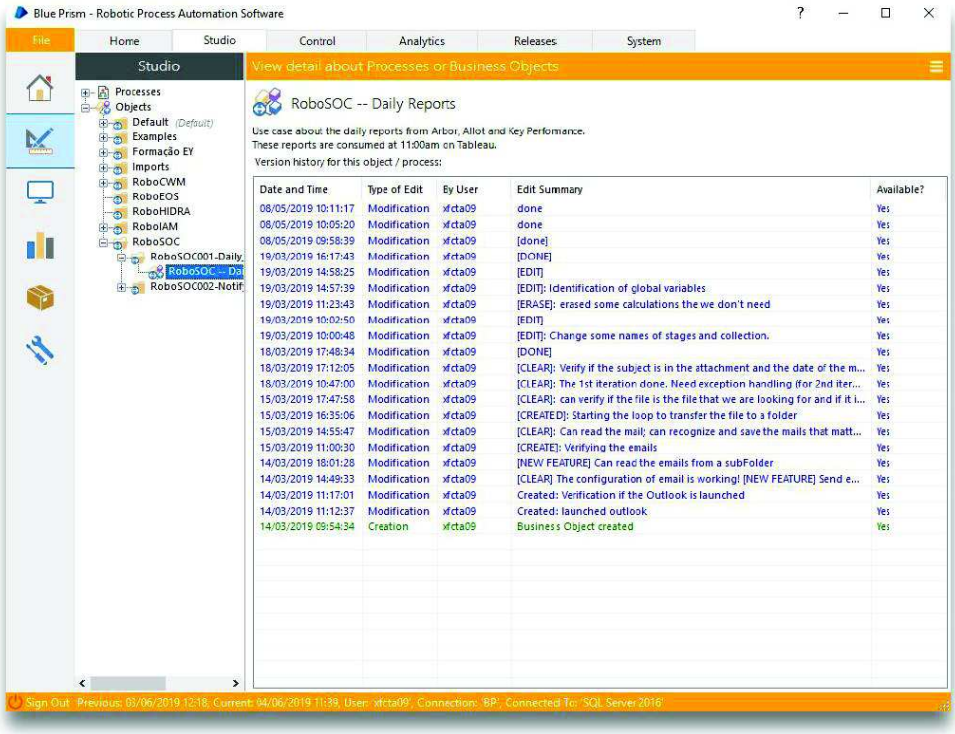


Figura 3.16: História de um *object*. Imagem retirada da ferramenta *Blue Prism*

3.4 *ATAR Labs*

O *ATAR Labs* é uma ferramenta focada para o ambiente SOC e utiliza o conceito SOAR. Esta ferramenta foi desenvolvida na Turquia, enquanto a empresa possuía o "status" de *start-up*. Contudo, tem vindo a ser aperfeiçoada nos últimos anos.

3.4.1 Objetivo do *ATAR Labs*

A finalidade do ATAR é mitigar ataques que sejam feitos à rede da empresa. O SIEM monitoriza a rede através de regras implementadas pela equipa SOC. Estas regras nomeiam o que é considerado o comportamento normal de um utilizador. Numa situação que um utilizador esteja dentro da rede e o seu comportamento se desvie das regras, o SIEM gera automaticamente um alerta e é aberto um *ticket*.

No caso da MEO, a rede é de grande escala e devido à sua grandeza são gerados vários alertas diariamente. Devido ao número elevado de alertas gerados por dia, há necessidade filtrar os falsos positivos[21] existentes entre eles. Atualmente esta filtragem é feita pelos analistas SOC que se regem a partir das características do alerta. Durante a análise, se nenhuma característica define o

[21] Comportamentos considerados ataques mas após a sua análise determina-se como comportamentos normais.

comportamento anormal, os analistas tomam como um falso positivo e fecham o *ticket*. Acontece que esta análise é feita repetidamente, consumindo tempo aos analistas. Para além disso, quando são encontrados verdadeiros positivos, os analistas consomem mais tempo devido à mitigação.

Como o número de ataques que se sucedem diariamente é elevado, muito tempo é "desperdiçado" em processos que se tornam repetitivos e temporalmente consumistas que impedem os analistas SOC de tratar os ataques de maior relevância. O propósito do ATAR é analisar estes ataques repetitivos e mitigá-los com ou sem intervenção dos analistas. Desta forma, é possível aos analistas dispensarem mais tempo na análise de ataques de maior severidade.

3.4.2 Funcionamento do *ATAR Labs*

O processo de mitigação de incidentes no *ATAR Labs* é feito através do contacto direto com os SIEMs existentes na empresa (*i.e.*, integram-nos) e com o sistema de *ticketing*.

Sempre que é gerado um alerta no SIEM, o *ATAR* reage. Após a receção dos incidentes, é feita uma consolidação e análise das suas características. Estas duas fases são implementadas pela equipa SOC e o *ATAR* faz a filtração dos falsos positivos.

Caso o incidente analisado corresponda a algum caso de consolidação que a equipa SOC implementou, é considerado um verdadeiro positivo. A partir desse momento a ferramenta prossegue com a mitigação do ataque – este processo de mitigação também é implementado pela equipa SOC e é denominado por *playbook*[22]. Caso contrário, se o incidente não corresponder a nenhum caso de consolidação, significa que o incidente é um falso positivo. O *ticket* desse incidente é automaticamente fechado pelo ATAR e a ferramenta tratará do próximo incidente.

Nos dias de hoje há dois SIEMs na MEO: o *Micro Focus ArcSight* e o *AlienVault*. Contudo, entrará um novo SIEM: *QRadar* da IBM que será o único a estar em funcionamento no futuro. Por este motivo, todos os clientes existentes no *ArcSight* e no *AlienVault* estão em fase de migração para o *QRadar*. Para além disso, na tentativa de integrar o *AlienVault* no *ATAR*, foi descoberto um *bug*[23] e de momento não existe nenhum SIEM integrado com o SOAR.

Os casos de uso propostos iniciam-se todos com *e-mails* e estes não são incidentes, mas sim notificações de um incidente. Mesmo que sejam implementados *parsers* para os *e-mails* recebidos, não está a ser dado o uso apropriado à ferramenta. Por outro lado, os casos de uso propostos não terminam com processos de mitigação mas sim com processos de notificação ou processos simples de *business*.

Apesar de existir uma ideia base parecida entre as tecnologias SOAR e RPA onde se poupa tempo aos trabalhadores para realizar tarefas de maior importância, estas tecnologias são para finalidades distintas. O SOAR serve para tarefas a nível de SOC, que filtra os falsos positivos e mitiga os ataques de menor esforço (a nível de análise). Enquanto o RPA serve para tarefas

[22] Corresponde ao conjunto de ações executadas sobre o ataque de modo a mitigá-lo

[23] Este *bug* levará tempo a ser corrigido porque os próprios *developers* do *ATAR* não reconhecem o motivo do *bug* nem onde se situa.

repetitivas a nível de *business*. Visto que o RPA é mais abrangente e os casos de uso apresentados são mais compatíveis ao RPA, não haverá nenhum caso implementado no *ATAR Labs*.

3.5 Análise final das ferramentas de RPA

Sabendo que a única tecnologia possível para a concretização do projeto foi o RPA, houve a possibilidade de explorar o *Automation Anywhere* e o *Blue Prism* empiricamente. A Tabela 3 apresenta algumas das características que as ferramentas possuem e tem o objetivo de resumir as diferenças entre elas.

Característica	***Automation Anywhere***	***Blue Prism***
Função *Record*	✓	✗
Interação com aplicações	✓	✓
Flexibilidade nos *browsers*	✗	✓
Modo de desenvolvimento	Pseudo-código	*Flowchart*
Criação de VBOs	✓	✓
Dashboards	✓	✓
Scheduler	✓	✓
Papéis dos clientes	✓	✓
Auditorias	✓	✓
Gestão de credenciais	✗	✓
Flexibilidade dos robots	✗	✓

Tabela 3: Análise das ferramentas de RPA após sua utilização

De acordo com a Tabela 3, o *Blue Prism* não possui a funcionalidade *Record*. De acordo com [8], David Moss explicou que a funcionalidade *Record* não foi implementada para evitar atalhos a quem desenvolve os robots. Para além disso, quando se faz uma gravação de um conjunto de comandos, este conjunto é estático e por esse motivo é adicionada a característica "Flexibilidade dos robots" da Tabela 3. O *Automation Anywhere* tem a funcionalidade *Record*, contudo, falha na flexibilidade. Ou seja, torna-se um *trade-off*. Outro ganho da *Blue Prism* é o facto de ter uma componente dedicada somente às credenciais. Onde guarda os *usernames* e as *passwords* de forma segura. No caso do *Automation Anywhere*, para fazer *login* de uma plataforma *web*, tem que se inserir as credenciais diretamente no robot e não obtê-las externamente.

Deste modo conclui-se que ambas são boas ferramentas, contudo o utilizador deve fazer uma avaliação relativamente às prioridades como por exemplo segurança, rapidez de implementação e flexibilidade dos robots.

Capítulo 4
Casos de Uso

Neste capítulo são apresentados os casos de uso do projeto. Cada caso será explicado detalhadamente, indicando o seu objetivo e como é realizado pela equipa SOC. Também é apresentada uma proposta da lógica que será implementada para cada caso de uso.

4.1 *Daily Reports from ArcSight to Tableau*

Neste caso de uso são utilizadas três aplicações e todas elas estão interligadas: o *Micro Focus ArcSight*, o *Allot* e o *Arbor*. Atualmente, o *ArcSight* é um dos SIEMs existentes no SOC da MEO. O *Allot* e o *Arbor* são as aplicações responsáveis por mitigar ataques de *Denial of Service* na rede da empresa. Atualmente, a execução das aplicações efetua-se da seguinte forma: quando há um evento com um comportamento semelhante a um de DoS, o *Allot* é o primeiro a detetar e iniciar o processo de mitigação. De seguida, o *Arbor* deteta o mesmo ataque e inicia o seu processo de mitigação. Ou seja, apesar da diferença e independência entre as aplicações, ambas têm o mesmo propósito. Atualmente, o *ArcSight* gera diariamente os dois relatórios (com extensão CSV) seguintes:

- O relatório "[Statistics] [Arbor & Allot]" tem a lista de todos os incidentes detetados – e mitigados – pelo *Allot* e *Arbor* do dia anterior. Cada incidente é composto pelas seguintes características: o seu ID (o identificador de cada incidente é um inteiro não cifrado), a sua duração, a categoria do evento, o país de onde vem o ataque (determinado através do IP da máquina), entre outras. A produção deste relatório está programada para as 07h00 da manhã no *ArcSight*;
- O relatório "[Statistics] [CSIRT Key Performance Indicator]", produzido às 08h00 da manhã, resume-se à listagem de *Key Performance Indicators* – que inclui uma avaliação dos incidentes reportados no dia anterior através de medidas criadas pela equipa SOC. Neste relatório existem três resumos:
    - O primeiro resumo tem como base o risco[24], o dispositivo onde ocorreu o incidente e o número total de incidentes nesse dispositivo (exemplo apresentado na Tabela 4);

Risk	Device Vendor	Device Product	Total
Low	*ArcSight*	*ArcSight*	*14 703*

Tabela 4: Caso exemplo do primeiro resumo no relatório de *Key Performance Indicators*

    - O segundo resumo organiza-se primeiro pelo nome do cliente, o risco, o tipo de incidente[25] e o número total de incidentes que correspondam às três componentes. É ilustrado um exemplo deste resumo na Tabela 5;

[24]O risco está divido em três níveis: *Low*, *Medium* e *High*. Estes níveis estão definidos a partir de regras criadas pela equipa SOC.

[25]O incidente pode ser de três tipos: *Base*, *Correlation* e *Aggregated*. Estes tipos estão definidos a partir de regras criadas pela equipa SOC.

Customer	Risk	Type	Total
Sara	*Medium*	*Aggregated*	*3 255*

Tabela 5: Caso exemplo do segundo resumo no relatório de *Key Performance Indicators*

- O terceiro resumo (exemplo na Tabela 6) lista todos os incidentes com base nas três componentes mencionadas nos dois primeiros resumos: cliente, risco e dispositivo.

Customer	Risk	Device Vendor	Device Product	Total
Sara	*Medium*	*ArcSight*	*ArcSight*	*5 064*

Tabela 6: Caso exemplo do terceiro resumo no relatório da *Key Performance Indicators*

Nos exemplos ilustrados pelas tabelas Tabela 4, Tabela 5 e Tabela 6, poder-se-ia fazer uma leitura da seguinte forma:

- Tabela 4: Houveram 14 703 ataques de risco *Low* onde tanto o fornecedor e o produto são *ArcSight*;
- Tabela 5: O cliente Sara sofreu no total 3 255 ataques do tipo *Aggregated* com risco M*edium;*
- Tabela 6: O cliente Sara sofreu no total 5 064 ataques com risco *Medium* em que o fornecedor e o produto são o *ArcSight*.

Estes relatórios são gerados para serem processados noutra aplicação (não pertencente ao SOC): *Tableau*. O *Tableau* trata-se de uma aplicação de *Business Intelligence* onde todos os documentos processados são para fins analíticos. Após o processamento dos documentos, estes apresentam-se graficamente por forma a facilitar a análise. Deste modo, estes relatórios servem para uma análise diária e concluir em que condições está a rede da empresa.

A Figura 4.1 representa o caso de uso *Daily Reports from ArcSight to Tableau* concretizado pelos analistas SOC e apresenta-se em forma de *System Sequence Diagram* (SSD)[26]:

- O *ArcSight* produz o primeiro relatório (*Arbor* e *Allot*) às 07h00 da manhã e envia um *e-mail*, com o relatório anexado, para a *Inbox* dos analistas SOC. O envio do relatório representa-se pela função *SendReportofArbor&Allot(report)*;
- O *ArcSight*, às 08h00 da manhã, produz o relatório correspondente ao *Key Performance Indicators* e envia para a *Inbox* dos analistas SOC o *e-mail* com o relatório anexado. A função correspondente é *SendReportofKeyPerformance(report)*;
- Considerando a situação normal, o analista SOC verifica a *Inbox* no *Outlook*. Este comportamento representa-se pela função *SeeOutlookInbox()*;
- Se o analista SOC encontrar os relatórios que procura, extrai-os. Este passo representa-se na função *GetReports()*;
- Quando o analista SOC possuir os relatórios, transfere-os para uma pasta partilhada associada ao *Tableau* para serem processados. Esta transferência está representada na função *TransferReportsToFolder(reports)*.

[26] SSD é uma representação que formaliza as descrições relacionadas a um caso de uso que ilustra, através de pequenos passos, a troca de informação entre atores (no contexto do trabalho são os analistas e o robot) e os sistemas envolvidos no caso de uso [46]

Este caso de uso deve ser concretizado entre o momento em que os dois relatórios estão na *Inbox* até às 11h00 da manhã. Isto deve-se à hora de processamento programada no *Tableau*.

Apesar da simplicidade do caso de uso, este tem que ser concretizado diariamente. Se o analista SOC tiver a *Inbox* cheia de *e-mails* de outros conteúdos (que não sejam os relatórios), vai despender algum tempo à procura dos *e-mails* com os relatórios. Por isso, verifica-se que é uma tarefa que consome tempo ao analista.

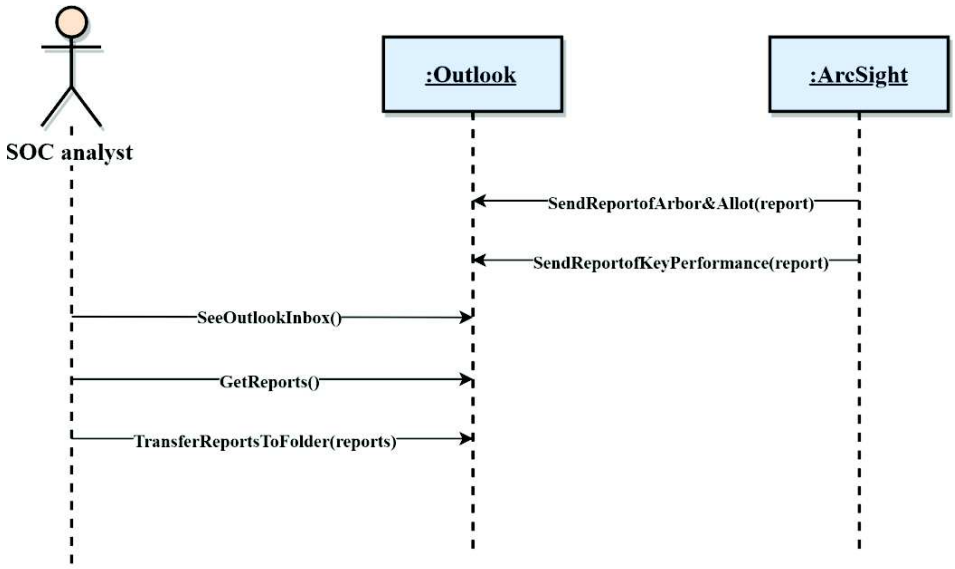


Figura 4.1: SSD que representa a execução do caso de uso *Daily Reports from ArcSight to Tableau*

O comportamento do robot será semelhante ao comportamento do analista SOC. Porém, o robot não tem a mesma capacidade de decisão, pois não reconhece sozinho quais os *e-mails* a extrair. Para tal, é necessário criar regras no robot. A implementação do robot apresenta-se (em alto-nível) sob a forma de SSD na Figura 4.2.

Neste caso de uso, as duas primeiras interações são iguais às do SSD na Figura 4.1. O robot terá a sua própria *Inbox* e os relatórios vindos do *ArcSight* passam a ser enviados para o *e-mail* do robot. A interação do robot com o *Outlook* é diferente da do analista, ilustrada na Figura 4.2, e realiza-se da seguinte forma:

- O robot configura-se ao *Outlook* através de um servidor interno da MEO ao qual o *e-mail* do robot está associado, inserindo as suas credenciais. Esta etapa ilustra-se na função *ConfigureToOutlook(username, pwd)*;
- Em seguida, o robot verifica a *Inbox* e extrai todos os *e-mails* para uma *collection* – estrutura de dados do *Blue Prism*. Estes dois passos representam-se, respetivamente, pelas funções *SeeOutlookInbox()* e *GetEmails()*;
- A *collection* é iterada, onde todos os *e-mails* são verificados, conforme o *loop* representado na função *VerifyPropertiesForEachMail()*, com base em três propriedades:

- *Subject*: O *e-mail* com o relatório do *Arbor* e *Allot* inicia sempre com o título "[Statistics] [Arbor & Allot]" e o *e-mail* com o relatório do *Key Performance Indicators* tem sempre o título "[Statistics] [CSIRT Key Performance Indicator]". O robot irá procurar estas *strings* no *subject* de cada *e-mail*;
- *Sender*: a entidade que envia estes *e-mails* tem que ser sempre o *ArcSight*. O robot terá que verificar se o *sender* é o correto;
- *Attachments*: o robot tem que verificar se o *e-mail* tem algum ficheiro em anexo e, se esse anexo é o esperado (em cada *e-mail* só pode existir um anexo). Os relatórios relacionados com o *Arbor* e *Allot* têm no começo do seu nome "[Statistics] [Arbor & Allot]" e, os relatórios relacionados com

os *Key Performance Indicators* têm sempre o título "[Statistics] [CSIRT Key Performance Indicator]". O robot irá procurar por estes nomes nos anexos.

- Para qualquer circunstância em que uma destas propriedades não seja verificada, o robot descarta automaticamente o *e-mail* e passa para o próximo *e-mail* na *collection* – se existir.
- Após a verificação das propriedades (esta verificação representa-se na função *PropertiesAreSatisfied()* que devolve um *boolean*) e se todas elas corresponderem (*boolean* igual a *True*), o robot obtém o anexo e transfere-o para a pasta partilhada para processamento no *Tableau*. As operações descritas representam-se nas funções *GetReportAttached()* e *TransferReportToSharedFolder(report)*, respetivamente;
- Por fim, para o robot não voltar a verificar os mesmos *e-mails*, o *e-mail* atual é transferido para uma subpasta da *Inbox* denominada de "Processed". Este passo apresenta-se a função *TransferEmailToProcessedFolder(email, name)*.

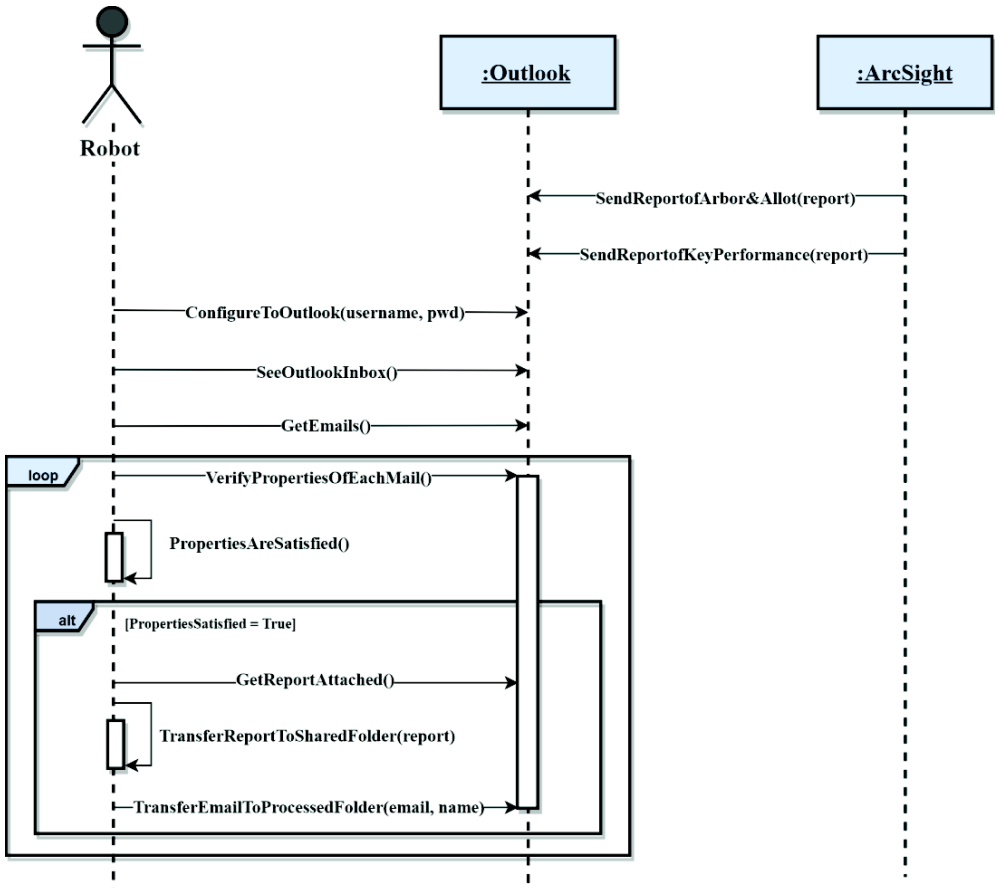


Figura 4.2: Representação da implementação em alto-nível do robot

4.2 *Notify Customers of DoS Attacks with Reports*

A MEO, tendo o seu SOC, vende o serviço de ciber segurança a clientes externos. O departamento de ciber segurança da MEO defende as redes dos clientes, porém não os notifica dos ataques. Os ataques analisados são de *Denial of Service* e são investigados na aplicação *Arbor*. A motivação do caso de uso *Notify Customers of DoS Attacks with Reports* é prestar aos clientes um serviço completo na proteção das suas máquinas (mitigar os ataques e notificar os clientes da ocorrência dos mesmos).

Quando um ataque ocorre, o *Arbor* envia um alerta ao *ArcSight*. O *ArcSight* gera um alerta e envia um *e-mail* para o sistema de *ticketing* da MEO (RTIR) gerando um *ticket*[27]. Quando o *ticket* é gerado, os analistas SOC recebem notificações vindas do *Arbor* – *e-mail* e sms – com informações do ataque e o *ticket* associado. Quando o ataque é mitigado e o *ticket* fechado, os analistas são informados que o ataque terminou. A realização atual do caso de uso é *event driven*, ou seja, sempre que um ataque inicia e o *Arbor* o reconheça, o analista é informado e notifica o cliente. Após o seu término, o analista SOC recebe a nova informação e envia um novo *e-mail* ao cliente a notificá-lo do fim do ataque.

De momento, existe apenas um único cliente que tem este serviço a pedido do mesmo. O caso de uso subdivide-se em dois subprocessos que passam a descrever de seguida:

- Notificação DoS do começo do ataque ao analista SOC (Figura 4.3):

- O *Arbor* deteta o ataque e envia um *e-mail* para a *Inbox* dos analistas SOC sobre o começo do ataque. Neste *e-mail* estão as informações do ataque (por exemplo, o ID no *Arbor*) e o URL do *ticket* correspondente. Esta etapa associa-se à função *SendEmailOfStartNotification(ticket, ID)*;
- Após a receção do *e-mail*, o analista lê-o e copia o *ticket* e o ID correspondentes ao ataque para pesquisar. Estes passos correspondem, respetivamente, às funções *SeeEmail()* e *GetTicketAndIDInEmail()*;
- O analista abre o *Arbor* e pesquisa o ataque através do ID guardado e regista as características do ataque (como por exemplo, a hora de começo, o IP *source* do ataque, o IP de destino, entre outros). Estas ações correspondem, respetivamente, às funções *GoToIncident(ID)* e *GetPropertiesOfIncident()*;
- Após a obtenção das características do ataque, o analista cria um *e-mail* ao cliente com o ID e as características do ataque – *CreateNewEmail(ID, Customer, PropertiesList*). Após a criação do *e-mail*, envia-o – *SendStartEmailToCustomer()*.

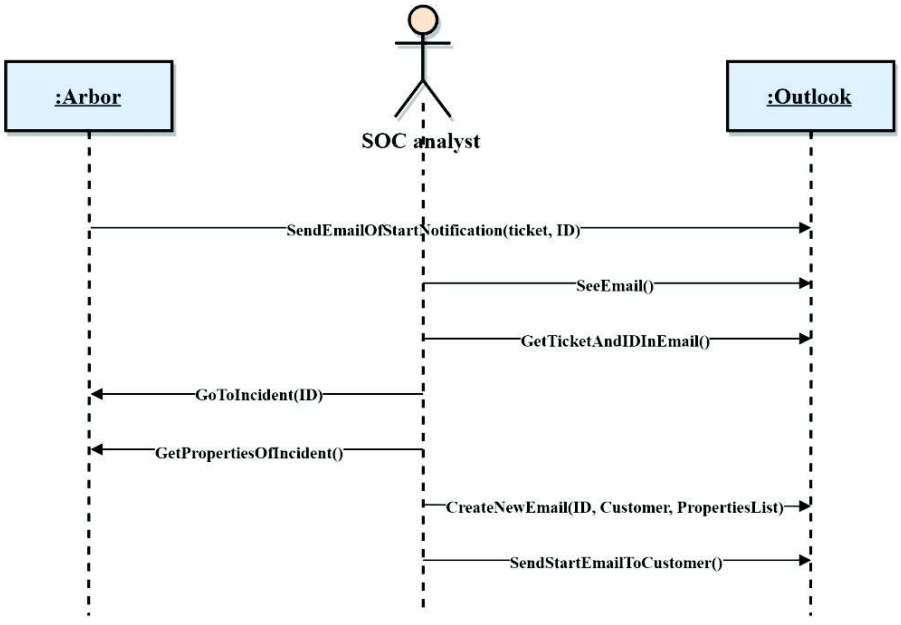


Figura 4.3: SSD que representa a execução caso de uso *Notify Customers of DoS Attacks with Reports*

[27] Os *tickets* gerados são para ataques DoS que tenham somente severidade *High*

- Quando é enviada uma notificação do término do ataque ao analista SOC (Figura 4.4):

Quando o ataque é mitigado, o *Arbor* envia um *e-mail* para a *Inbox* dos analistas SOC informando-os do fim do ataque. Neste *e-mail* estão as informações do ataque (por exemplo, o ID no *Arbor*) e envia o URL do *ticket* correspondente. Este comportamento está associado à função *SendEmailOfEndNotification(ticket, ID)*;

Após a receção do *e-mail*, o analista SOC vê o *e-mail* recebido e através do mesmo, guarda o *ticket* e o ID correspondentes ao ataque. Estes passos correspondem, respetivamente, às funções *SeeEmail()* e *GetTicketAndIDInEmail()*;

O analista SOC abre e pesquisa no *Arbor* o ataque através do seu ID previamente adquirido, e guarda as suas características atualizadas. Estas ações correspondem, respetivamente, às funções *GoToIncident(ID)* e *GetUpdatedPropertiesOfIncident()*;

Por fim, o analista SOC cria o *e-mail* com todas as características atualizadas, com o ID do ataque do *Arbor* e o *e-mail* é enviado para o cliente. As ações mencionadas representam-se nas funções *CreateNewEmail(ID, Customer, PropertiesList*) e *SendEndEmailToCustomer()*, respetivamente.

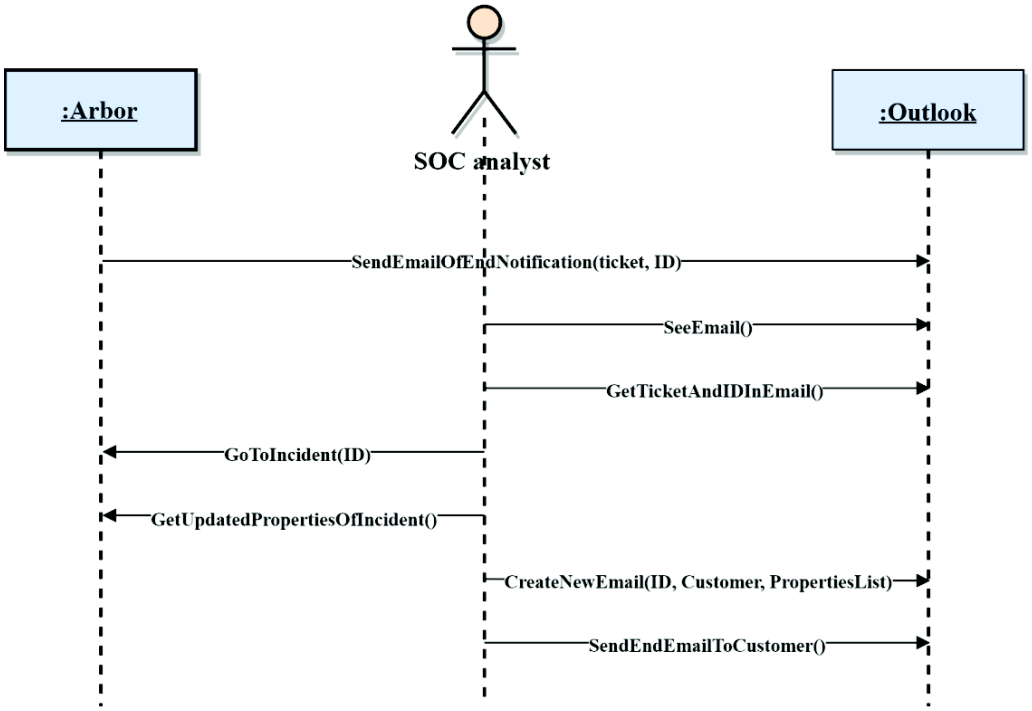


Figura 4.4: SSD que representa a execução do segundo subprocesso do caso de uso *Notify Customers of DoS Attacks with Reports*

A criticidade deste caso de uso é alta, visto que dá conhecimento aos clientes de que estão a ser protegidos.

Apesar de o processo decorrer com base *event driven*, de momento não é possível uma máquina – *i.e.*, robot – detetar instantaneamente a chegada de novos *e-mails* à *Inbox*. Devido a este contratempo, ficou estabelecido que o robot verá, a cada hora, se existem novos ataques. Para ter garantias de que todos os ataques são verificados pelo robot, é feita a pesquisa dos últimos 100 ataques no *Arbor*, cada uma destas pesquisas pode resultar em uma das três alternativas:

- **O ataque ter iniciado, mas ainda não terminado**: *i.e.*, o ataque teve início mas ainda não terminou. Neste caso, é enviado um *e-mail* para o respetivo cliente para informar que teve início um ataque *DoS*. No *e-mail* inclui a informação apresentada na página *web* do *Arbor* e é anexado o PDF correspondente do ataque;
- **O ataque ter iniciado e terminado (deteção via regras do robot)**: *i.e.*, o ataque já tinha dado início e o robot notificou o cliente com um *e-mail* inicial. Agora o robot envia um e-mail a notificar o cliente de que o ataque terminou. No corpo do *e-mail* estará a informação atualizada relativa ao ataque e anexa-se o novo PDF;
- **O ataque foi iniciado e terminado (sem o robot o ter detetado)**: *i.e.*, o robot verificar que o ataque já iniciou e terminou sem estar registado. Neste caso, envia-se um *e-mail* para o cliente a notificar de que um ataque iniciou e finalizou. No corpo do *e-mail* é inserida a informação completa do ataque anexando o PDF (produzido pelo *Arbor*) correspondente ao mesmo.

Para tentar cobrir os últimos 100 ataques DoS aos clientes, o caso de uso *Notify Customers of DoS attacks with Reports* está ilustrado na Figura 4.5 decorrendo da seguinte forma:

- O robot inicia o *browser* já com a página de *login* do *Arbor* aberta inserindo as suas credenciais. Estes passos representam-se pelas funções *OpenBrowserAndArbor()* e *LoginArbor(username, pwd)*;
- O robot segue para a página que lista os últimos ataques DoS e extrai as características dos ataques para uma *collection* – este procedimento ilustra-se nas funções *GoToIncidentsPage()* e *GetIncidents*(*100*);
- O robot irá usar o *e-mail* para notificar os clientes dos ataques extraídos, de forma que abre o *Outlook*. Este passo representa-se pela função *OpenOutlook(username, pwd)*;
- A *collection* dos ataques possui o registo das características de cada ataque como por exemplo o ID no *Arbor*, o tempo de início e de fim (caso tenha terminado), o cliente que está a ser atacado, entre outras. Através das características guardadas, o robot verifica se o ataque já foi totalmente tratado ou não – esta verificação representa-se na função *SeeTypeOfEmailToWrite()* que devolve o tipo de *e-mail* na variável *Type*;
- A *collection* é iterada através do ID, e se o ataque não foi totalmente tratado, então robot direcciona-se à sua página (*GoToIncidentPage(ID)*) e faz *download* do relatório do mesmo (*GetIncidentReport()*). Caso o ataque esteja totalmente tratado, o robot itera para o próximo ID;
- De seguida, é criado um novo *e-mail* para o respectivo cliente. Este *e-mail* tem um *Body* inicial específico onde são adicionadas as características do ataque. O relatório é anexado e quando o *e-mail* estiver criado, envia-se. Estas etapas representam-se, respectivamente, nas funções *WriteEmail(Type, ID, Customer, Body, Report)* e *SendEmail()*. Quando o *e-mail* é enviado, o robot regista o ataque numa estrutura própria do *Blue Prism* chamada *Queue*;
- Quando forem iterados os 100 ataques, o robot faz o *logout* no *Arbor* e fecha as aplicações utilizadas. Estes passos representam-se pelas funções *LogoutArbor()*, *CloseBrowser()* e *CloseOutlook()*.

Apesar de serem vistos os últimos 100 ataques, nada garante ao robot que estes ataques terminem antes de serem inseridos outros 100. Sendo assim, para garantir cobertura total aos clientes, após a execução do processo apresentado na Figura 4.5 deve ser executado outro processo onde o robot vai consultar e guardar a informação dos ataques registados e pendentes – *i.e.*, que ainda não foram totalmente tratados.

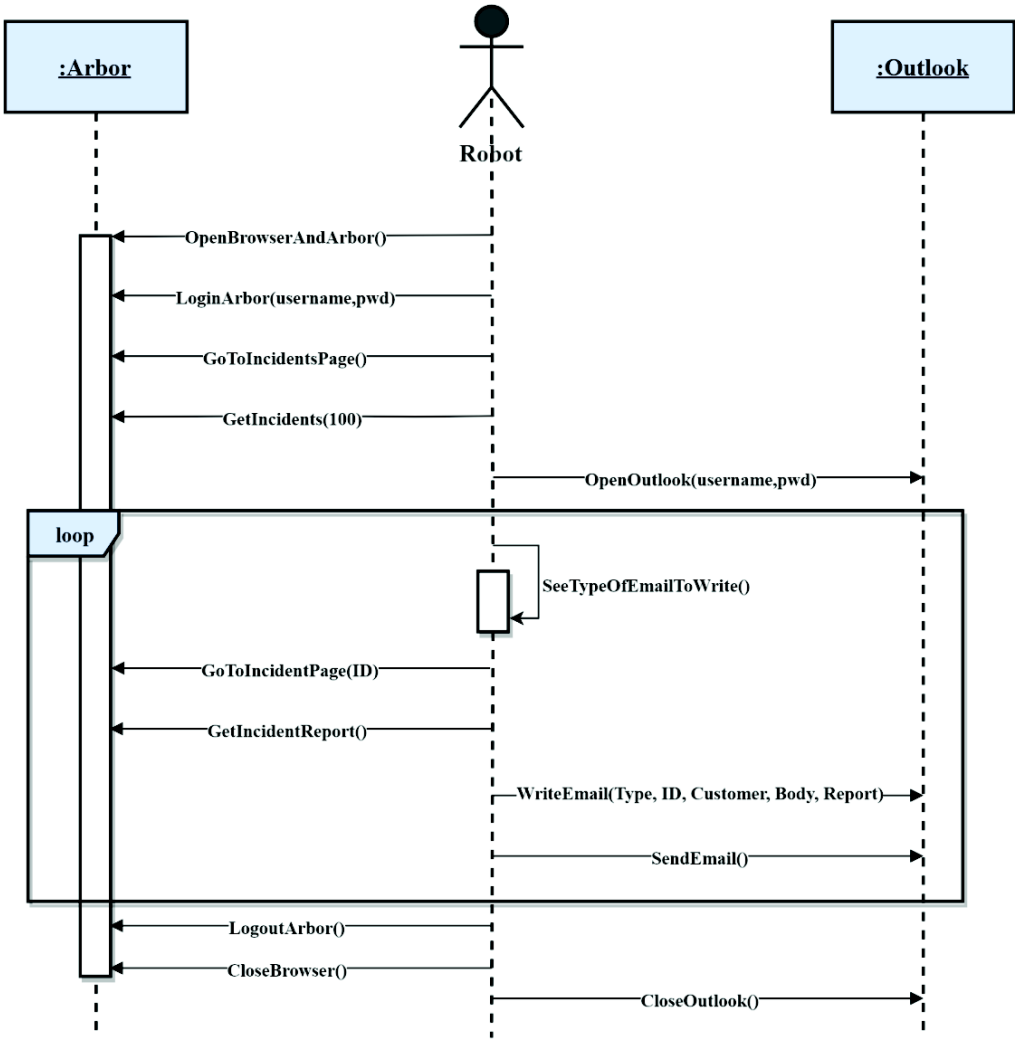


Figura 4.5: SSD que representa uma proposta de implementação do caso de uso *Notify Customers of DoS Attacks with Reports*

Estes ataques serão iterados e pesquisados no *Arbor* para saber o seu estado de terminação. O processo chama-se *Verify Pendent DoS Attacks* e apresenta-se na Figura 4.6 da seguinte forma:

- As funções *OpenBrowserAndArbor()*, *LoginArbor(username, pwd)*, *GoToIncidentsPage()* e *OpenOutlook(username, pwd)* são executadas nas mesmas condições que no processo representado pela Figura 4.5;
- O robot através da *Queue* obtém os ataques pendentes (juntamente com as características) e guarda-os numa *collection*. Esta *collection* é iterada através do ID, e na página que são listados os ataques de DoS, o ID do ataque é pesquisado. Estas etapas representam-se pelas funções *GetAllPendentIncident(QueueName)* e *SearchIncidentByID(ID)*;
- De seguida, o robot extrai as características da página *web* e compara-as com as características guardadas na *collection* – ilustrada na função *IncidentEnded()* que devolve um *boolean IncidentEnded*.

    a) Se estas se mantiverem iguais (*IncidentEnded* = *False*), então o ataque ainda não terminou e o robot passa para o próximo;

b) Caso contrário, o robot direciona-se à página do ataque (*GoToIncidentPage(ID)*) e faz *download* do relatório do ataque (*GetIncidentReport()*);

- O *e-mail* final é escrito pelo robot com o ID do ataque, um *Body* inicial específico e o relatório anexado. Após o desenvolvimento do *e-mail*, este é enviado para o respetivo cliente. Estas etapas representam-se pelas funções *WriteFinalEmail(ID, Customer, Body, Report)* e *SendEmail()*;
- Por fim, as funções *LogoutArbor()*, *CloseBrowser()* e *CloseOutlook()* executam-se nas mesmas condições apresentadas na Figura 4.5.

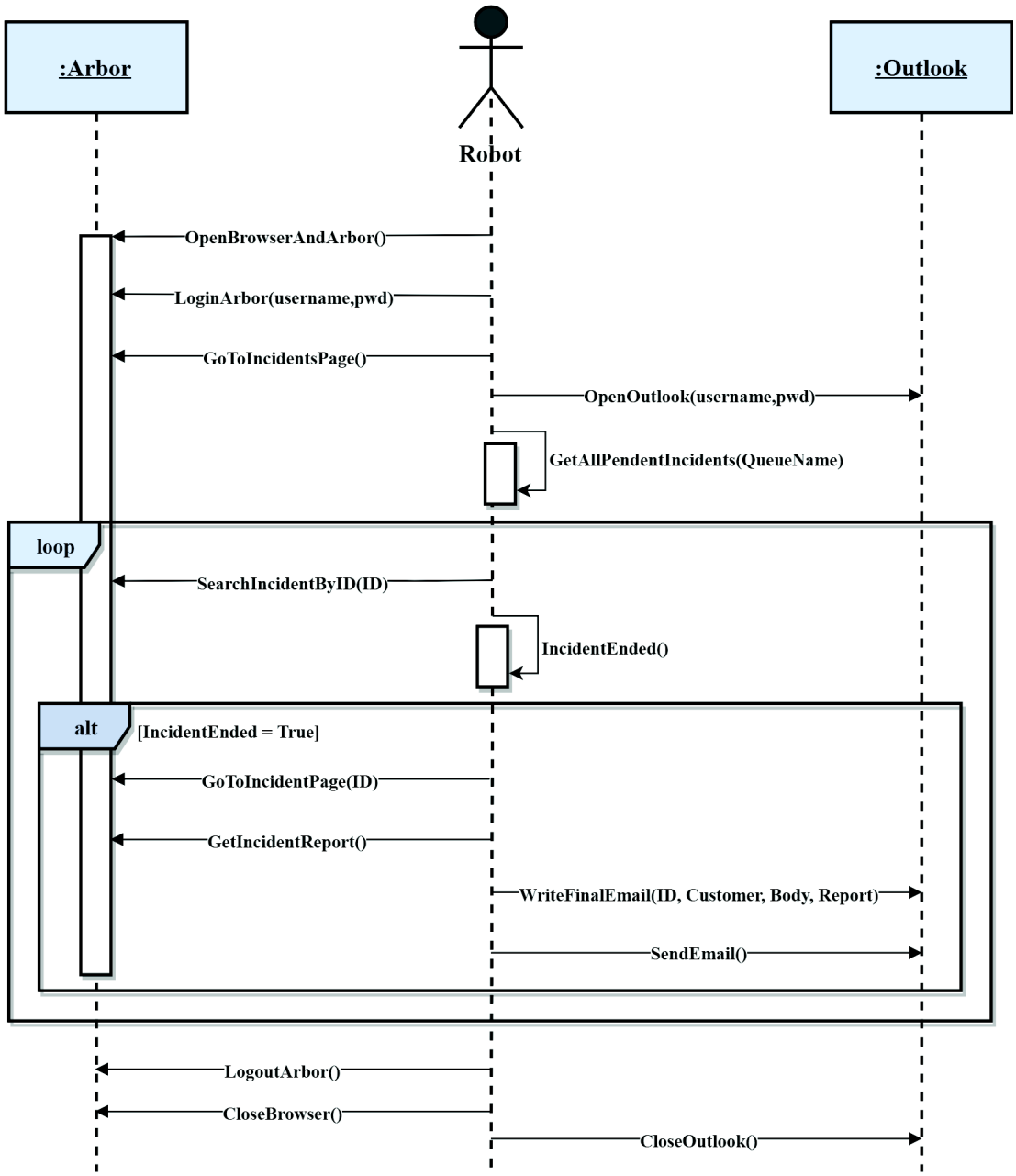


Figura 4.6: Representação do processo *Verify Pendent DoS Attacks* sob a forma de SSD

4.3 *Verify brute-force Attacks in LoginPT*

A rede da MEO é uma rede de grande escala que proporciona várias plataformas e serviços aos colaboradores – *i.e.*, trabalhadores – e clientes. A rede é monitorizada pelos SIEMs *Micro Focus ArcSight* e *AlienVault*. O *AlienVault* monitoriza todas as plataformas e serviços da MEO e é o SIEM de interesse para este caso de uso.

Para aceder às plataformas e serviços da MEO, tanto os colaboradores como os clientes precisam de fazer o *login*. Por uma questão de facilitismo, foi pensado num único *login* para cada cliente (ou colaborador) para aceder a todos os serviços. Este *login* é conhecido como *LoginPT* e é monitorizado pelo *AlienVault*. Sempre que um utilizador falhe dez acessos consecutivos, o *AlienVault* despoleta um alerta. Atualmente, o analista SOC recebe um *e-mail* com o *ticket* do alerta, notificando-o da possibilidade da ocorrência de um ataque.

Os sistemas usados neste caso de uso são: *AlienVault*, *RTIR*, *Excel* e *HP Service Manager (HPSM)*. Devido à diversidade de aplicações usadas, o caso de uso *Verify brute-force Attacks in LoginPT* é explicado em três partes. A primeira parte (*Get Incident CSV*) representa-se pela Figura 4.7:

- Quando é detetado um comportamento semelhante a um *brute-force attack*, o *AlienVault* gera o alerta e envia um *e-mail*, representado pela função *SendAlertEmail()*, para a caixa de correio dos analistas SOC;
- O analista, após a receção do *e-mail*, verifica-o e copia o ID do ataque correspondente no RTIR. Estes dois passos correspondem às funções *SeeEmail()* e *GetIncidentID()*, respetivamente;
- Através do ID copiado, com a função *GoToIncidentPage(ID)* o analista pesquisa o ID respetivo ao ataque e extrai o ficheiro em formato CSV. A extração equivale à função *GetCSV()*;
- Visto que o *Outlook* não será mais usado no caso de uso, a aplicação é fechada através da função *CloseOutlook()*.

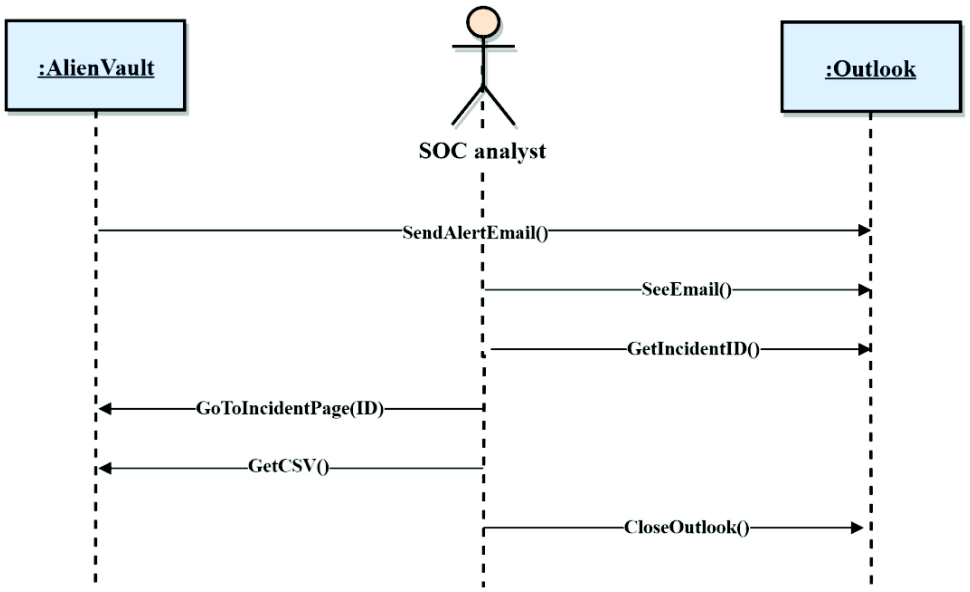


Figura 4.7: Representação da parte *Get Incident CSV* do caso de uso *Verify brute-force Attacks in LoginPT*

A segunda parte (*Create Pivot Table and Insert in RTIR*) do caso de uso e os sistemas presentes nesta parte são o *Microsoft Excel* e *RTIR*. A parte *Create Pivot Table and Insert in RTIR* é ilustrada na Figura 4.8:

- Com o ficheiro CSV extraído, o mesmo é aberto na aplicação *Excel* (representado na função *OpenExcel(CSV)*);
- Tendo os dados relativos ao ataque, estes estão sumarizados no ficheiro e o analista cria uma *Pivot Table*[28] onde são criadas regras para apresentação da mesma. Estes dois passos estão representados pelas funções *CreatePivotTable()* e *SetBoundaries()*;
- De seguida, o próprio analista faz a análise da *Pivot Table* para saber se os dados correspondem a um verdadeiro positivo ou não. Esta análise corresponde à função *Analize(PivotTable)*;
- Após a análise, a *Pivot Table* é guardada e a aplicação é fechada. Estes passos correspondem às funções *Save()* e *CloseOutlook()*;
- Com o ID ainda obtido no *e-mail* e referenciado na Figura 4.7, o analista vai ao sistema de *ticketing* (RTIR) e faz o *login* com a função *Login(username, pwd)*. Depois pesquisa pelo ataque através do ID. Este passo corresponde à função *SearchTicket(ID)*;
- Na sumarização do *ticket*, o analista insere a *Pivot Table* (*InsertPivotTable(CSV)*) e faz *logout* do sistema (*Logout()*).

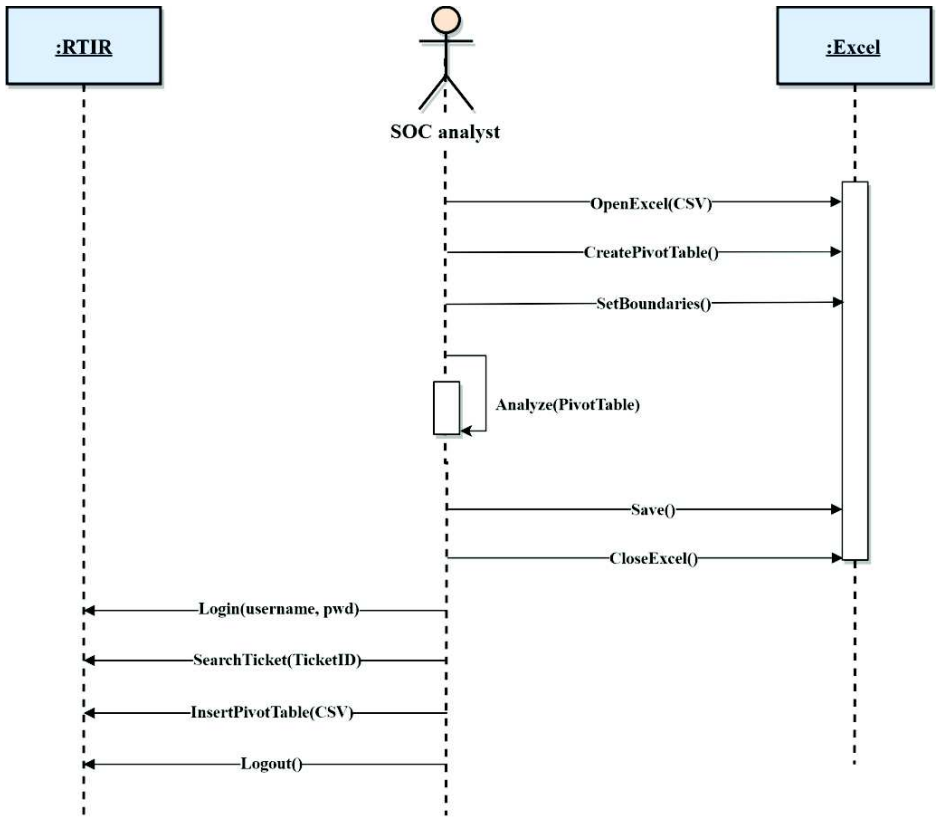


Figura 4.8: Representação, sob a forma de SSD, da parte *Create Pivot Table and Insert in RTIR* do caso de uso *Verify brute-force Attacks in LoginPT*

[28] A *Pivot Table* é uma estrutura dinâmica que permite a análise simples de um grande conjunto de dados [17]

A terceira parte chamada *Create Ticket in HPSM* limita-se à criação de um *ticket* no sistema *HP Service Manager* (HPSM) e notifica uma equipa externa do ataque. Esta parte ilustra-se na Figura 4.9:

- Para concluir o caso de uso, o analista faz o *Login(username, pwd)* na aplicação *HPSM* e inicia a criação do *ticket* com os dados correspondentes ao ataque. Este passo representa-se pela função *CreateTicket()*;
- Após a finalização da criação do *ticket*, o analista encaminha a *Pivot Table*, por *e-mail*, a uma equipa externa do ataque e esta tarefa representa-se pela função *SendPivotTable(email, CSV)*;
- Por fim, o analista faz *logout* da aplicação (*Logout()*).

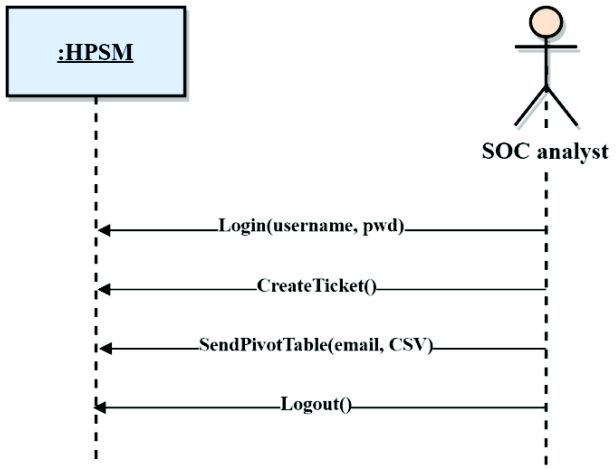


Figura 4.9: Representação, sob a forma de SSD, da parte *Create Ticket in HPSM* do caso de uso *Verify brute-force Attacks in LoginPT*

Este caso de uso sempre foi executado deste modo devido à ausência de uma alternativa para este tipo de ataques. A maior desvantagem da execução deste processo (apresentado pelas figuras 4.7, 4.8 e 4.9) é o facto de ser lenta. Isto dá vantagem ao atacante (caso o alerta seja um verdadeiro positivo), pois oferece-lhe a oportunidade de acertar na *password* e ter acesso aos serviços. Com o *software ATAR Labs*, é possível garantir uma alternativa para este processo tornando-o mais rápido pois através das informações dadas no SIEM *AlienVault*, o *ATAR Labs* faz a correlação das informações e se corresponderem a um verdadeiro positivo, mitiga automaticamente (por regras criadas pelos analistas) o ataque.

Deste modo, o caso de uso será implementado no *ATAR Labs*. A solução para este caso de uso no *ATAR Labs*, será bloquear temporariamente o sujeito na *firewall*. Esta solução é mais rápida e impede o atacante de tentar mais vezes aceder à conta. Contudo, devido aos motivos apresentados na Secção 3.4.2, a implementação deste processo será adiada até o ambiente SOC ter as condições ideais para usar o *software* SOAR.

4.4 *Notify Customers of DoS Attacks with Screenshots*

Após a realização do caso de uso descrito na Secção 4.2 os requisitos foram alterados devido à identificação de informação sensível (relacionada com a rede interna da MEO) nos relatórios do *Arbor*, de forma que a lógica e os objetivos se mantivessem os mesmos. As aplicações usadas são as mesmas: *Micro Focus ArcSight* e *Arbor*.

Sendo assim, a estrutura lógica mantém-se igual: duas notificações (uma de início e uma de fim) por cada ataque *DoS* e dois *e-mail*s de notificação do analista para o cliente. Contudo, devido à informação sensível existente nos relatórios, os analistas organizaram uma alternativa. A alternativa ponderada pela equipa SOC foi obter *screenshots* da aplicação com a informação necessária ao cliente. Deste modo,

foram tomadas as providências necessárias na composição dos *e-mails* de notificação aos clientes. O caso de uso *Notify Customers of DoS Attacks with Screenshots* atualizado (para notificações do começo de um ataque) ilustra-se na Figura 4.10:

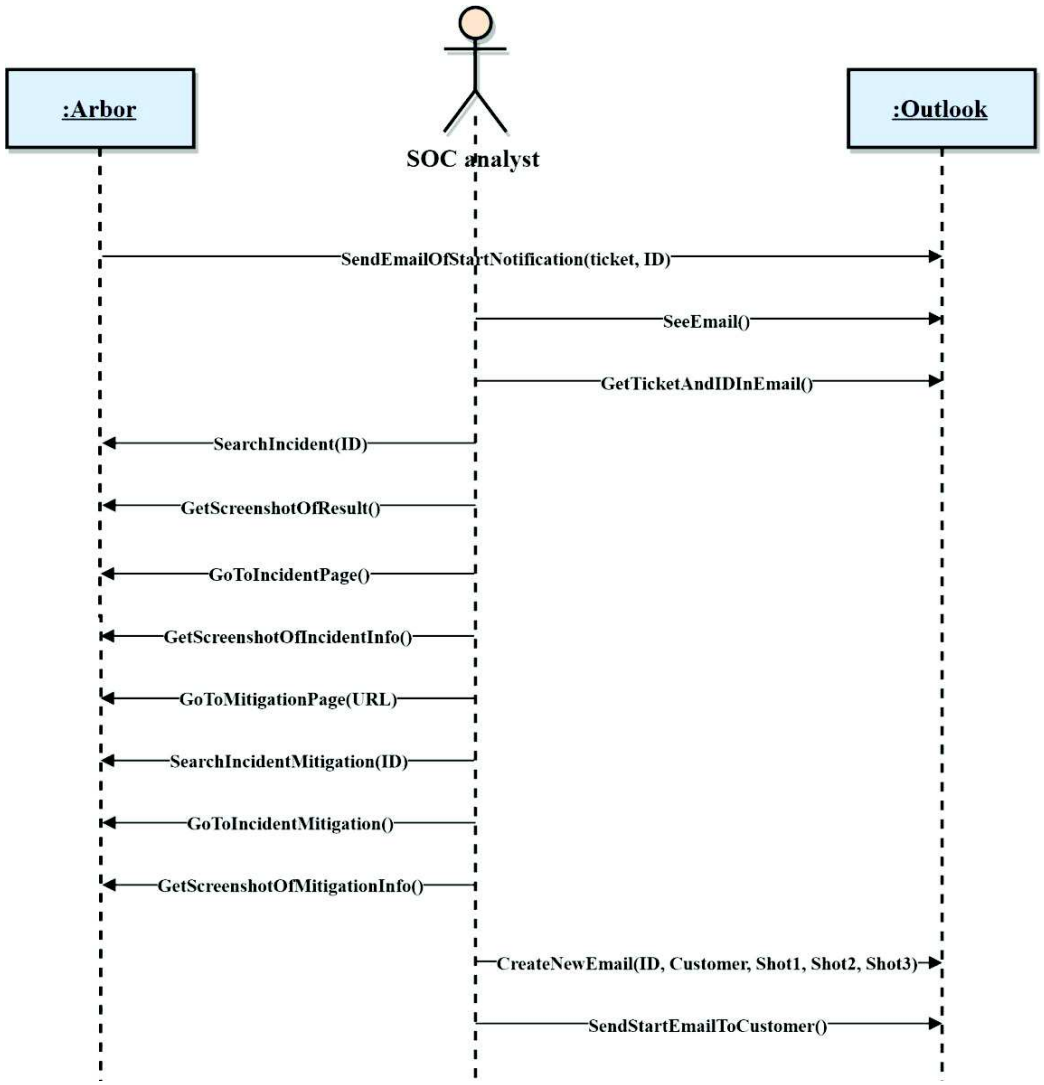


Figura 4.10: SSD que representa a execução do caso de uso *Notify Customers of DoS Attacks with Screenshots*

- O *Arbor* deteta o ataque e envia um *e-mail* para a *Inbox* dos analistas SOC sobre o começo do ataque. Neste *e-mail* estão as informações do ataque (por exemplo, o ID no *Arbor*) e o URL do *ticket* correspondente. Esta etapa associa-se à função *SendEmailOfStartNotification(ticket, ID)*;
- Após a receção do *e-mail*, o analista SOC lê-o e copia o *ticket* e o ID correspondentes ao ataque para pesquisa. Estes passos correspondem, respetivamente, às funções *SeeEmail()* e *GetTicketAndIDInEmail()*;
- Com o ID copiado do *e-mail*, este é pesquisado no *Arbor*. Sendo o ID único, haverá sempre um único resultado que se identifica na função *SearchIncident(ID)*. Exemplifica-se um resultado desta pesquisa na Figura 4.11 que possui informações "base" do ataque (por exemplo, o cliente afetado, a severidade e duração). Após a obtenção do resultado, o analista faz um *screenshot* do mesmo com a função *GetScreenshotOfResult()*;

Figura 4.11: Exemplo de uma pesquisa de um ataque de *DoS* no *Arbor*

- Depois o analista vai à pagina do ataque (*GoToIncidentPage()*), e faz um *screenshot* com as informações relevantes para o cliente (*GetScreenshotOfIncidentInfo()*). Um exemplo das informações de um ataque para um cliente ilustra-se na Figura 4.12;

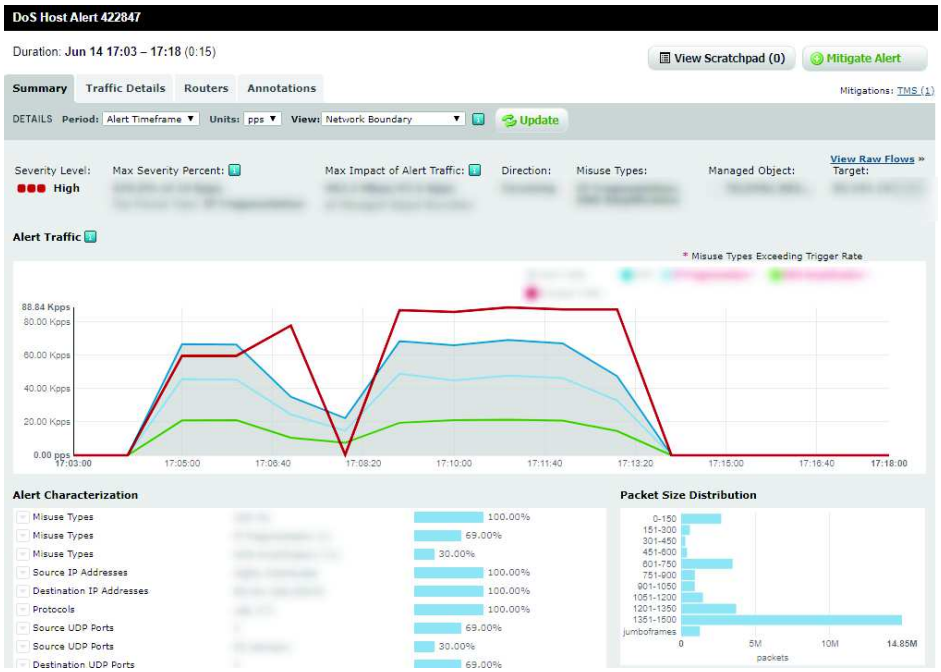


Figura 4.12: Informações detalhadas de um alerta no *Arbor*

- No caso de uso *Notify Customers of DoS Attacks with Reports* da Secção 4.2, os relatórios não possuíam informações relacionadas com a mitigação do ataque. Contudo, neste caso será enviada informação da mitigação num *screenshot*. O analista vai à página de mitigação (*GoToMitigationPage(URL)*) dos ataques no *Arbor* (como ilustrado na função *SearchIncidentMitigation(ID)*) e pesquisa pelo ID do ataque. Todavia a mitigação do ataque não é garantida resultando em dois caminhos:

    a) Caso exista mitigação do ataque, o analista carrega no resultado obtido, representado pela função *GoToIncidentMitigationPage()*;
    b) Caso contrário, o analista não apresenta nenhum *screenshot* de mitigação e a obtenção da informação termina neste passo.

- Na página do ataque mitigado (ou a ser mitigado) o analista faz um *screenshot* da informação necessária relacionada com a mitigação do mesmo. A Figura 4.13 ilustra um exemplo de um ataque mitigado. Este passo está representado na função *GetScreenshotOfMitigationInfo()*.

Figura 4.13: Exemplo da informação de mitigação de um ataque enviada ao cliente

- Por fim, o analista SOC cria o *e-mail* com todas as informações com o ID do ataque. Por fim, o *e-mail* é enviado para o cliente. Os comportamentos mencionados representam-se nas funções *CreateNewEmail(ID, Customer, Shot1, Shot2, Shot3*) e *SendEndEmailToCustomer()*, respetivamente.

O processo de término de um ataque procede nas mesmas condições apresentadas, porém a única função que diferencia é a primeira função da Figura 4.10. Em vez da função se chamar *SendEmailOfStartNotification(ticket, ID)* chama-se *SendEmailOfEndNotification(ticket, ID)* e quando o analista verificar o ataque, o *ticket* do mesmo já estará fechado.

Para que o procedimento apresentado seja realizado, o analista faz primeiramente uma análise crítica ao ataque. Apesar do nível de severidade ser *High* em todos os ataques analisados, o analista precisa de reconhecer se de facto a gravidade é suficiente para que o cliente seja notificado. Por esse motivo, o caso de uso tem pré-condições. Contudo, a motivação do desenvolvimento deste caso de uso é notificar os clientes dos ataques para lhes dar o conhecimento de que as suas máquinas estão protegidas permanentemente.

No caso de uso da Secção 4.2, não havia distinção entre os ataques (*i.e.*, todos os ataques eram verificados e notificados, sem qualquer restrição), contudo a implementação do robot deste caso de uso considera somente os ataques com nível de severidade *High*. A implementação é dividida em duas partes: a primeira parte compõe-se na obtenção dos ataques através da base de dados. Todos os ataques possuem um *ticket* e deste modo, a base de dados usada será do RTIR. A segunda parte é composta pela verificação dos elementos do ataque, obtenção das informações do mesmo e composição do *e-mail* a ser enviado ao cliente.

A primeira parte *Get Incidents from RTIR* apresenta-se na Figura 4.14, que executa da seguinte forma:

- O robot para aceder à base de dados do sistema de *ticketing*, precisa fazer uma conexão à mesma. Para isso, estabelece uma conexão através da função *Connect(server)*, à qual fornece o nome da base de dados como parâmetro;
- Após a conexão, o robot envia uma *query* do tipo *SELECT* para receber todos os ataques. A restrição da *query* é obter os ataques existentes dos últimos sete dias (garante que mesmo havendo um número elevado de ataques, há uma cobertura total) que está representado pela função *GetIncidents(query)*;

- O robot recebe da base de dados do RTIR a lista de todos os ataques (com informações dos mesmos) que sucederam nos últimos sete dias. Esta lista é iterada e em cada iteração, o ataque é inserido numa *Queue* (estrutura interna do *Blue Prism*) que guarda os novos incidentes. Este passo é representado pela função *InsertIncidentsToQueue(IncidentsList)*.

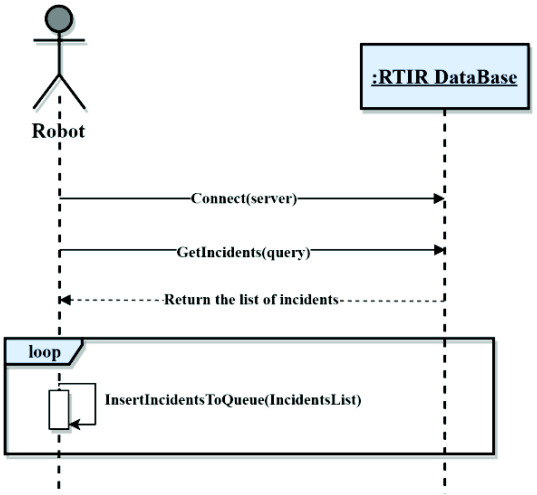


Figura 4.14: Representação sob a forma de SSD da parte *Get Incidents from RTIR* da implementação do robot do caso de uso *Notify Customers of DoS Attacks with Screenshots*

A segunda parte é ilustrada na Figura 4.15 sob a forma de SSD e executa no seguinte modo:

- As funções *OpenBrowserAndArbor()*, *LoginArbor(username, pwd)*, *GoToIncidentsPage()* e *OpenOutlook(username, pwd)* são executadas nas mesmas condições que no processo do caso de uso da Secção 4.2;
- Dentro da *Queue* estão guardados todos os ataques ocorridos nos últimos sete dias e esta é iterada pelo robot. Os ataques que não foram tratados passam pelo *loop* e o ID é pesquisado (representado pela função *SearchIncident(ID)*). De seguida é feito um *screenshot* do resultado (*e.g.*, Figura 4.11 e guarda como *Shot1*) e as informações obtidas são analisadas (*e.g.*, verificar se o ataque permanece aberto ou se já terminou). Estes dois passos estão ilustrados, respectivamente, pelas funções *GetFirstScreenshot()* e *VerifyIncident (IncidentInfo)*;
- Depois é selecionado o resultado da pesquisa do ataque e direcciona-se à página onde se situam as informações de maior detalhe e é feito o segundo *screenshot* (*e.g.* Figura 4.12 que guarda o *screenshot* como *Shot2*). Para constituição do *e-mail* são extraídas algumas informações como o ID, o tempo e o *source IP*. Por fim, é extraído o relatório do ataque que será guardado para a eventualidade de análises futuras. Estes passos estão representados pelas funções *ClickOnIncident()*, *GetSecondScreenshot()*, *GetSummaryInfo()*, *GetIncidentReport()*, respetivamente;
- Para a obtenção das informações da mitigação do ataque é inserido, diretamente, no *browser* o URL do site de mitigações do *Arbor* que se representa pela função *GoToMitigationPage(URL)* e o ID do ataque é pesquisado (representado pela função *SearchIncidentMitigation(ID)*);
- Quando é feita a pesquisa da mitigação do ataque, podem existir dois resultados:

    a) Não existirem quaisquer informações relacionadas com a mitigação do ataque e por esse motivo, não são extraídos mais dados do ataque;
    b) Caso contrário, é feito um *click* sobre o resultado que direcciona para a página da mitigação do ataque com todas as informações da mesma (*ClickOnIncidentMitigation()*). Na página de mitigação é feito o terceiro *screenshot* (e guarda o *screenshot* como o *Shot3*) e extraído o relatório com todos os dados da mitigação. Estes últimos dois passos estão ilustrados nas funções *GetThirdScreenshot()* e *GetMitigationReport()*.

- Por fim, quando são obtidas todas as informações do ataque, é criado um *e-mail* para o cliente correspondente que terá os três *screenshots* e um texto pré-elaborado com as informações obtidas pelo robot e o *e-mail* é enviado. Estas etapas estão descritas através das funções *WriteEmail(ID, Customer, Shot1, Shot2, Shot3)* e *SendEmail()*;
- Quando todos os ataques forem tratados, todas as aplicações a serem usadas são encerradas. O fecho das aplicações apresentam-se nas funções *LogoutArbor()*, *CloseBrowser()* e *CloseOutlook()*.

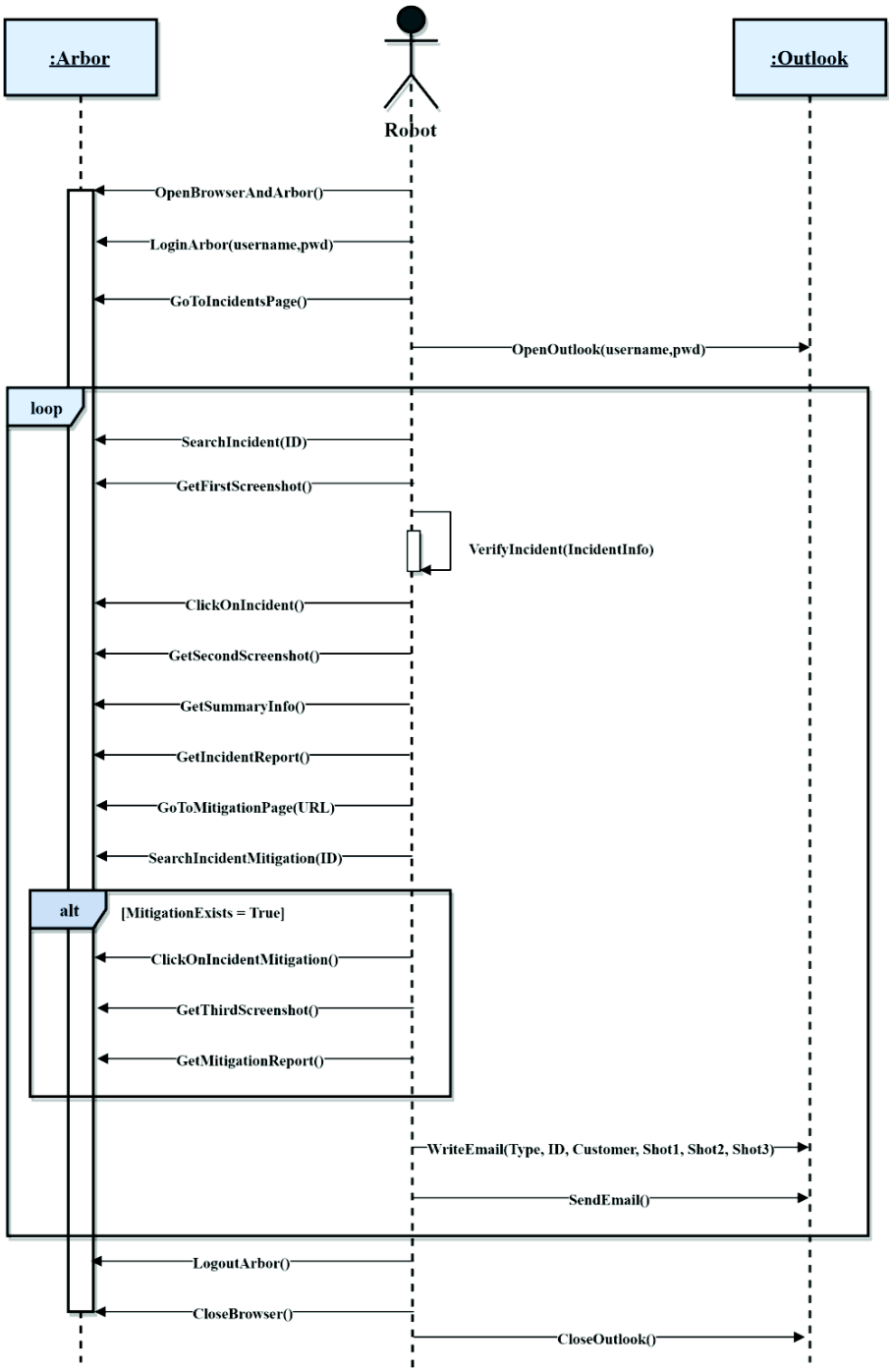


Figura 4.15:Representação sob a forma de SSD da segunda parte da implementação do caso de uso *Notify Customers of DoS Attacks with Screenshots*

Capítulo 5
Implementação dos casos de uso

Neste capítulo serão apresentadas as implementações dos casos de uso que foram aprovados para desenvolvimento e quais as dificuldades ultrapassadas.

5.1 *Daily Reports from ArcSight to Tableau*

Tal como mencionado na Secção 4.1, este caso de uso tem como finalidade mover relatórios que são enviados pelo *ArcSight*, diariamente, para uma pasta partilhada onde são processados (para fins analíticos) pelo *Tableau*.

A implementação deste caso de uso foi concretizada em duas iterações: a primeira iteração serviu para implementar a lógica (sem levantamento de exceções) e a segunda iteração foi para melhorar a lógica com o levantamento das exceções definidas.

5.1.1 Primeira Iteração

A lógica da implementação neste caso de uso foi realizada no *Object Studio* e depois colocada no *Process Studio* (Secção 3.3.2). Sendo que a aplicação usada neste caso de uso é o *Outlook*, foi necessário associar a aplicação ao *Application Modeller* para fazer *Launch* da mesma. A implementação foi dividida em duas partes, sendo a primeira parte iniciar a aplicação, e a segunda parte representa a lógica de extração e transferência dos relatórios. A primeira parte da lógica ilustra-se, sob a linguagem própria do *Blue Prism*, na Figura 5.1.

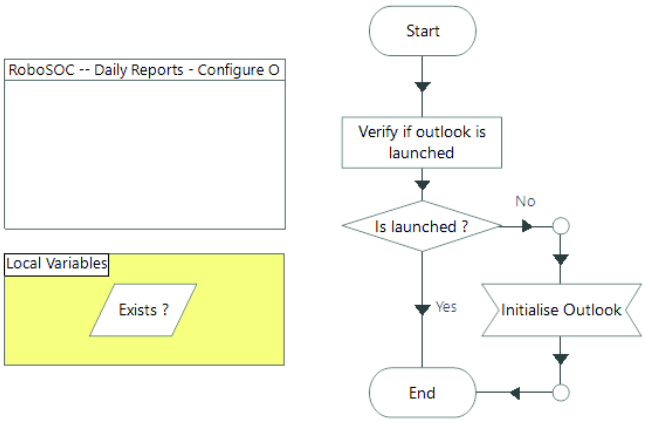


Figura 5.1: Implementação da primeira parte do caso de uso

Tal como apresentado na Figura 5.1, para fazer *Launch* da aplicação, é necessário verificar se está ativa – *i.e.*, iniciada. Para o caso da aplicação estar inativa, é feita uma ação para ativá-la. Caso contrário, mantém-se o ambiente da máquina nas mesmas condições para que o processo continue.

A segunda parte compõe-se pela extração dos relatórios e a sua lógica de extração apresenta-se na Figura 5.2.

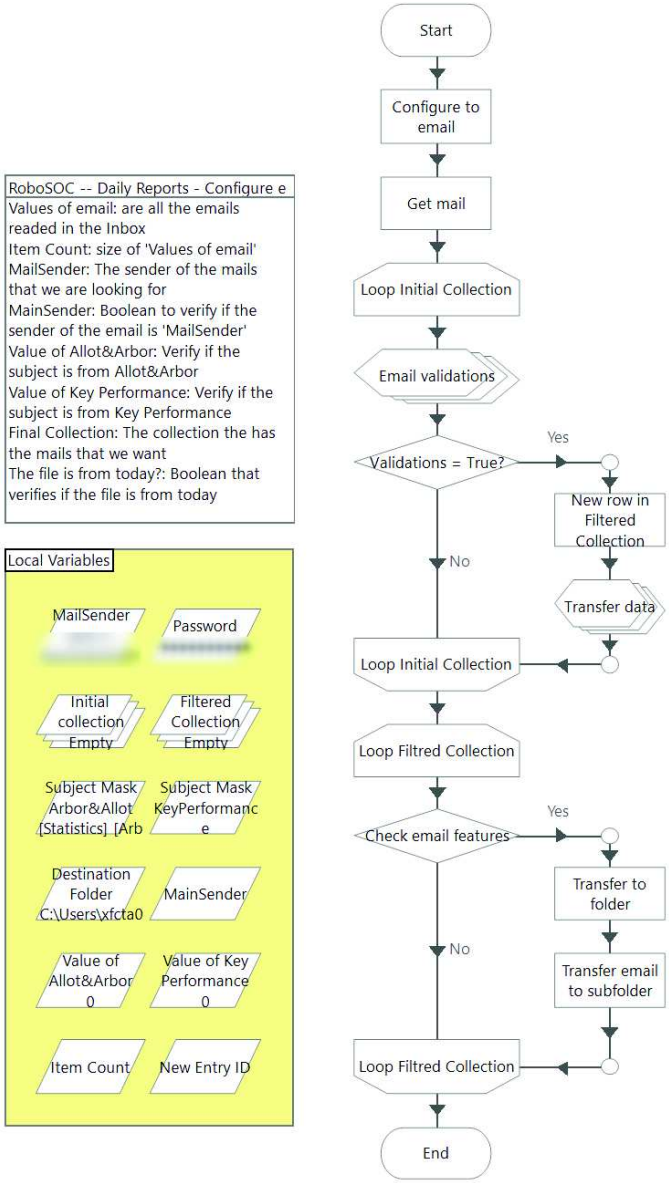


Figura 5.2: Implementação da extração dos relatórios nos *e-mails*

Tendo a aplicação ativa, é feita a configuração do *Blue Prism* com a *Inbox*. Após a configuração, são extraídos todos os *e-mails* e colocados na *collection Initial Collection*. Para detetar os *e-mails* procurados, estes são filtrados através da aplicação de duas validações de seguida explicadas:

- *Subject*: os títulos dos *e-mails* podem variar entre "[Statistics] [CSIRT Key Performance Indicator]" e "[Statistics] [Arbor & Allot]";
- *E-mail*: o *e-mail* só pode ser o respetivo ao *ArcSight*.

A *Initial Collection* é iterada e são feitas as validações. Quando as validações são respeitadas, todos os elementos desse *e-mail* são copiados para uma nova *collection* chamada *Filtered Collection*. De outro modo, é passado para o próximo *e-mail* da *collection* (caso haja mais *e-mails*).

De seguida, a *Filtered Collection* tem todos os *e-mails* de interesse e esta é iterada. Nesta etapa, é feita a validação do *attachment*. A nomenclatura dos *attachments* destes *e-mails* é: *Subject*_dd-MM-yy HH-mm. Sendo que as horas e minutos são da produção do relatório no *ArcSight*. Se os títulos dos mesmos respeitarem esta nomenclatura, então são válidos e são extraídos para a pasta partilhada do *Tableau*.

No *Process Studio* foi criado um processo que concretiza as duas etapas (*Launch Outlook* e *Extract attachments from emails*) das figuras Figura 5.1 e Figura 5.2 e ilustram-se na Figura 5.3.

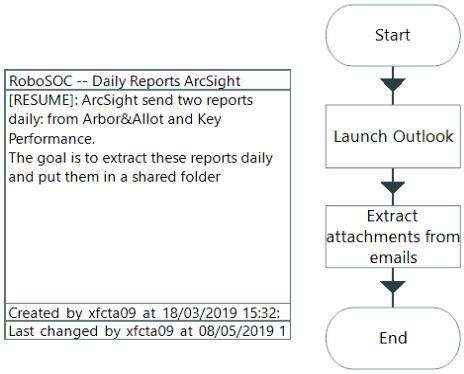


Figura 5.3: Implementação do processo final

5.1.2 Segunda Iteração

Nesta iteração a lógica foi simplificada – *i.e.*, são realizados menos passos para atingir o propósito – e foi inserida a gestão de erros. Com o ganho na experiência com o *Blue Prism*, foi possível implementar todo o processo no *Process Studio*, porque não há necessidade de ter o processo associado a alguma aplicação para o concretizar. Devido à ausência de uma associação a uma aplicação, não existe um *Application Modeller*. Esta abordagem traz algumas vantagens, como redução dos recursos necessários à execução do caso de uso, e por consequência, melhorando o desempenho.

Com a configuração direta entre o *Blue Prism* e o servidor, é possível obter os *e-mails* na *Inbox*. Por isso, a extração dos *e-mails* da *Inbox* decorre em *background*. O *Blue Prism* tem uma componente que permite guardar credenciais, tal como é apresentado na Figura 5.4, de forma segura e limita outros utilizadores (e/ou recursos) de usufruírem das mesmas. Na Figura 5.4 apresenta-se a página de criação de uma credencial e esta terá um nome representante. Na sua constituição está o *username*, a *password* (que no momento que é escrita, está automaticamente cifrada com AES-256 [18]) e é possível submeter uma data para a credencial expirar. Para além destes elementos é possível o cliente, que cria a credencial, escolher quais processos e máquinas têm permissão de obter a mesma, o que limita a sua acessibilidade

e impossibilita a sua escrita direta no processo. O processo inicia com a obtenção das credenciais para possibilitar a configuração entre o *Blue Prism* e o servidor de *e-mails*.

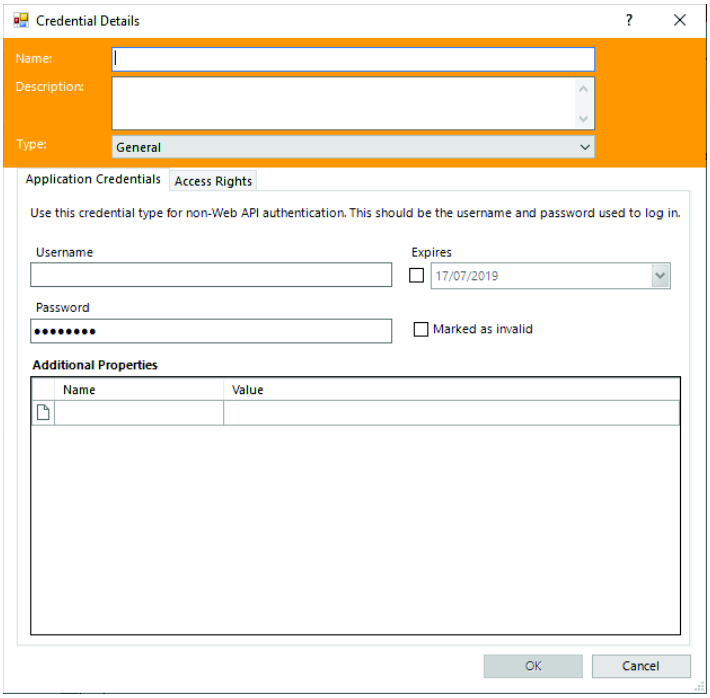


Figura 5.4: Janela de criação de uma nova credencial

A estrutura da lógica deste processo apresenta-se na Figura 5.5.

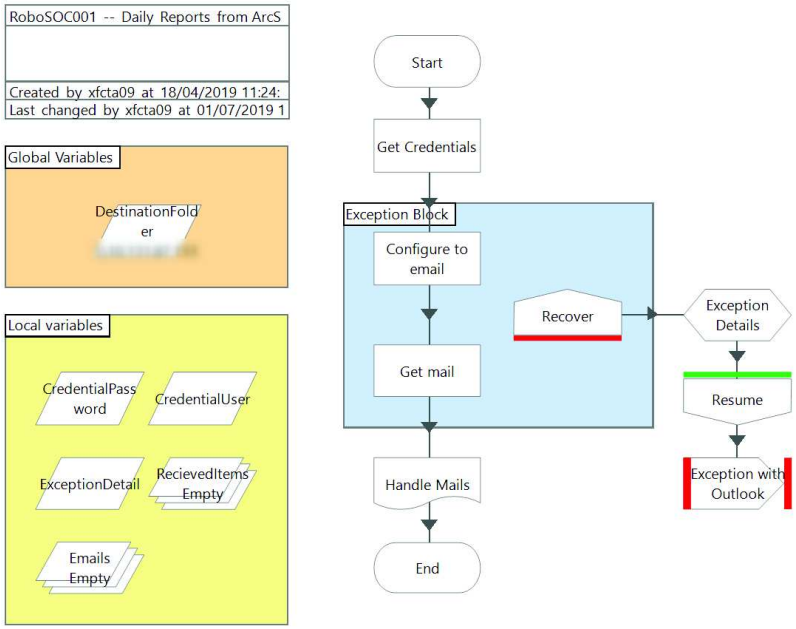


Figura 5.5: Estrutura da lógica do caso de uso *Daily Reports from ArcSight to Tableau*

Com a configuração e conexão estabelecida entre o *Blue Prism* e o servidor, é possível extrair os *e-mails* da *Inbox* e inseri-los numa *collection* chamada *ReceivedItems*. Esta *collection* é iterada dentro do subprocesso chamado *Handle Mails*. Este subprocesso apresenta-se na Figura 5.6.

O subprocesso *Handle Mails* resume-se à validação dos *e-mails* e a extração dos *attachments*. Ao contrário da primeira iteração, este processo tem somente uma *collection* a ser analisada e as validações são todas feitas simultaneamente. Sempre que as validações sejam respeitadas, o *attachment* do *e-mail* é extraído e guardado na pasta partilhada. Para impedir que o *e-mail* volte a ser processado, este é transferido para uma subpasta chamada *Processed*. Esta transferência também garante a *Inbox* estar sempre o mais vazia possível.

Relativamente à gestão de erros, há maior probabilidade de ocorrer exceções quando são feitas interações entre o *Blue Prism* e o servidor (ou seja, a configuração e extração dos *e-mails* na *Inbox*). Para impedir a interrupção do processo, estes passos foram envolvidos num bloco para levantar as exceções. Quando é levantada uma exceção, o *Blue Prism* passa automaticamente por uma fase de *Recover* onde são feitos os passos necessários para que o processo continue sem quaisquer erros. Os passos dependem sempre do caso de uso que está a ser implementado e em que ponto se situa.

Neste caso, são pedidos os detalhes da exceção para que se possa tomar conhecimento qual a exceção levantada e passa-se pela etapa *Resume* que retoma o *Blue Prism* o funcionamento normal.

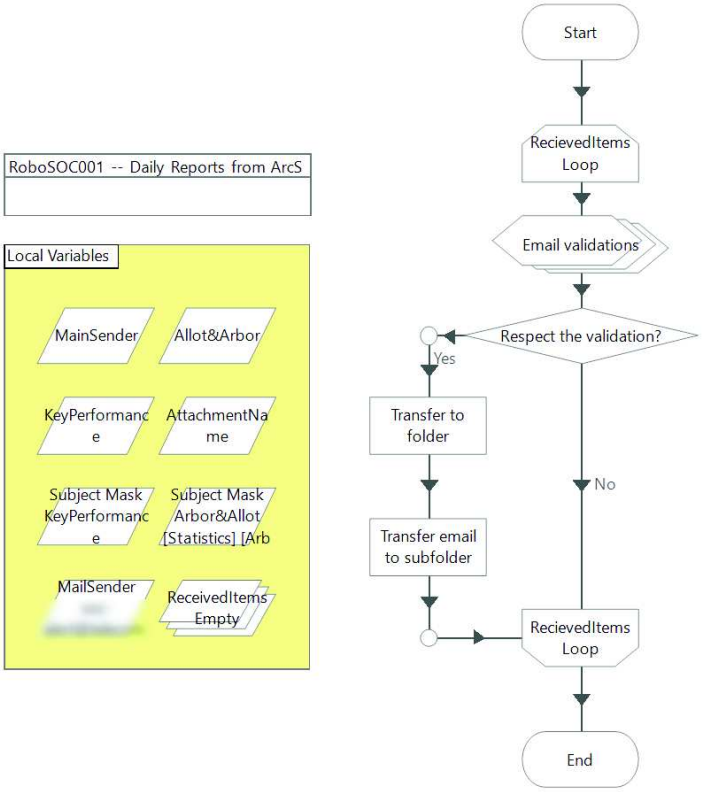


Figura 5.6: Implementação do subprocesso *Handle Mails*

5.2 *Notify Customers of DoS Attacks with Reports*

A implementação deste caso de uso foi feita com duas aplicações: o *Outlook* e o *Arbor*. O acesso ao *Arbor* é feito através de um *browser* (*Google Chrome*).

Esta implementação também foi feita em duas iterações e ambas foram feitas com os mesmos requisitos apresentados na Secção 5.1, ou seja, primeira iteração serviu para estruturar a lógica do caso de uso e a segunda iteração para o melhoramento inserindo o levantamento de exceções.

5.2.1 Primeira Iteração

Nesta iteração, o *Blue Prism* tinha a versão 6.4.0 e permitia interagir somente com *browser Internet Explorer* (IE). Por esse motivo, todo o caso de uso foi implementado com as aplicações *Outlook* e IE. A implementação realizou-se tanto no *Object Studio* como no *Process Studio* e foram criados múltiplos *objects* que, por fim, foram transferidos para um único *process*.

No *Object Studio*, a aplicação associada ao *Application Modeller* é o *Internet Explorer* (IE) e foi inserida uma regra que sempre que for lançado o *Launch*, a página da plataforma *Arbor* abre. Contudo, por motivos relacionados com os certificados da plataforma, o IE considera o *site* suspeito e por esse motivo devolve uma página inicial a indicar que não é de confiança. A página apresenta-se na Figura 5.7. O aparecimento desta página deve-se à falta de reconhecimento da entidade geradora do certificado. Por isso, o IE alerta o utilizador questionando-o se quer entrar no *Arbor*. Como esta página não é totalmente composta por elementos *HyperText Markup Language* (HTML) é impossível identificar alguns elementos. Como alternativa, usaram-se comandos que simulam o *mouse* e o teclado (este *object* apresenta-se na Figura A.1) para continuar o processo e passar para o *login*.

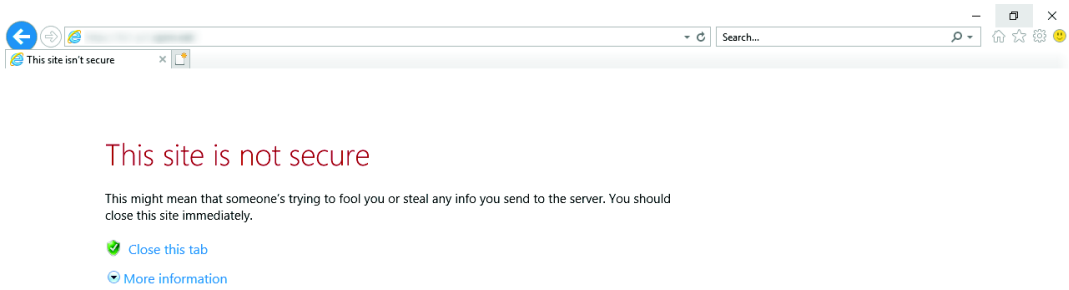


Figura 5.7: Página inicial quando o *Arbor* é aberto

Desde da página do *login*, todos os elementos das páginas são elementos HTML por isso a sua identificação é simples. No *login*, foram identificadas as barras correspondentes para inserção das credenciais e o botão para legitimar as mesmas. Este passo apresenta-se na Figura A.2.

Com a sessão iniciada, o robot direciona-se à página dos alertas de *Denial of Service* através de *clicks* nos elementos da página. Sempre que uma página de alertas é aberta, é apresentada uma tabela com dez alertas (todos com gravidades, clientes e dados distintos) e para garantir o tratamento de todos os alertas gerados, a tabela é redimensionada para apresentar 100 alertas. Quando se apresentam os 100 alertas, estes são extraídos e guardados numa *collection*. Estes passos apresentam-se nas figuras A.3 e A.4.

A gestão dos alertas apresenta-se na Figura 5.8 e a *collection* dos alertas é iterada. Neste *object*, é feita a verificação de cada alerta e o reconhecimento do tipo de *e-mail* a ser enviado. Nesta etapa há três *collections* (duas delas são externas ao *object* sendo variáveis globais):

- *Collection* dos alertas: contém os 100 alertas que são verificados se já foram tratados ou não;
- *Collection* dos clientes: contém os nomes dos clientes que estão no *Arbor* e o respetivo *e-mail*. A obtenção do *e-mail* do cliente está ilustrado na Figura A.5;
- *Collection* dos alertas tratados: é semelhante a um *log* que indica o ID do alerta, o tipo de *e-mail* que foi enviado, a data de envio e o cliente. Caso sejam enviados dois *e-mails* para o mesmo cliente (um a descrever o início de um ataque e outro o término do mesmo), são escritas duas linhas do

mesmo incidente dentro do *log*. Este *log* é um documento de extensão CSV (*Comma-Separated Values*) e é extraído como ilustrado na Figura A.6.

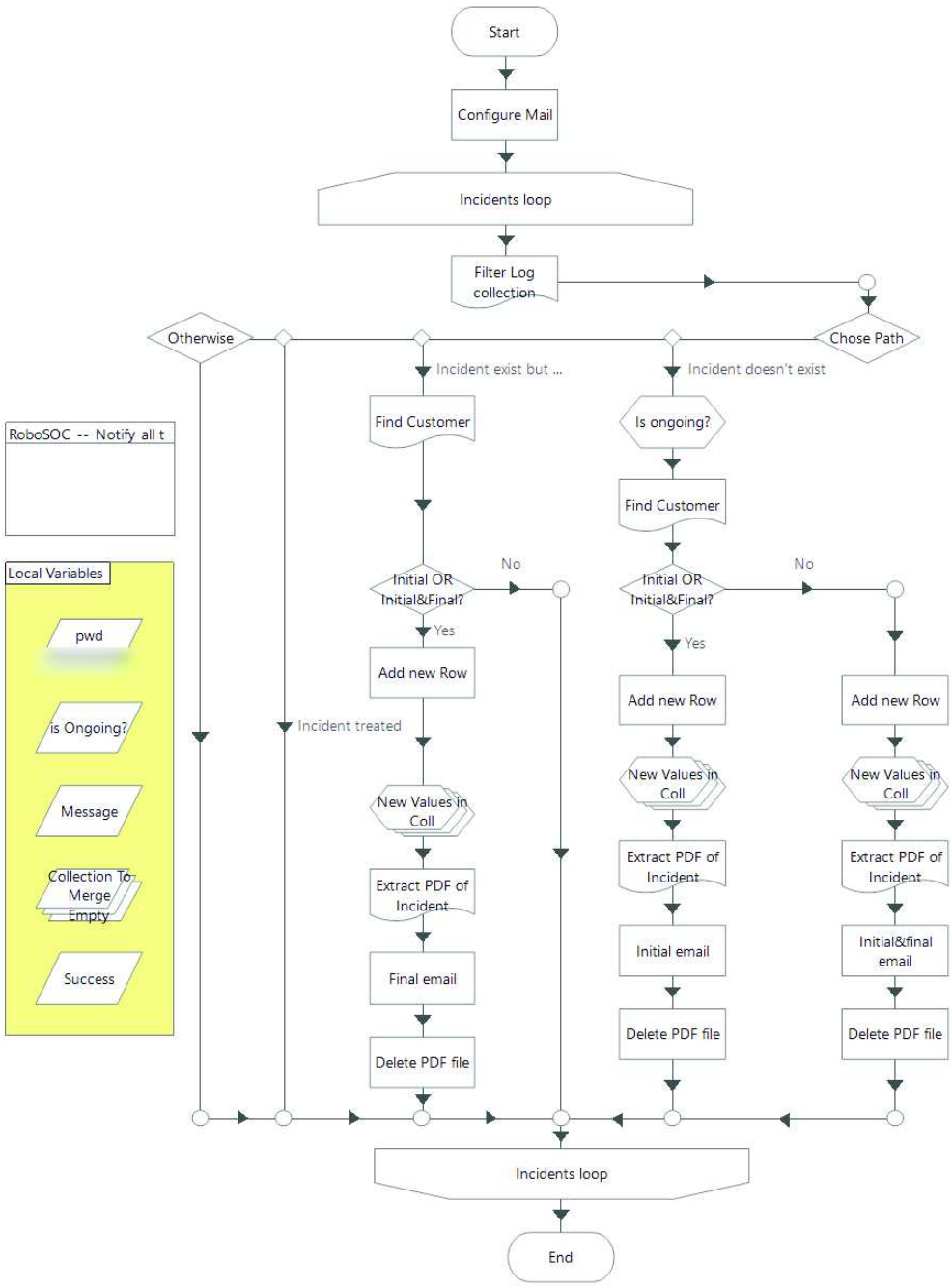


Figura 5.8: Fluxograma responsável pela gestão dos 100 alertas obtidos no *Arbor* (implementado no *Object Studio*)

As *collections* estão associadas entre elas. Para saber se os alertas foram tratados, a *collection* dos alertas é iterada. A Figura A.7 ilustra a *collection* dos alertas tratados que são filtrados através do ID. Nesta filtração podem existir três possibilidades:

- Não existe nenhuma linha no log: significa que o alerta não foi tratado até ao momento;

- Existe uma linha no *log* o que pode significar uma destas alternativas:
    a. Pode já ter sido enviado um *e-mail* a descrever somente o começo do ataque, se for, na linha estará escrito "Initial" ou;
    b. Pode já ter sido enviado um *e-mail* a descrever o começo e o término do ataque. Se for, estará escrito na linha "Initial&Final";
- Existem duas linhas no *log*: significa que o alerta já foi totalmente tratado e o robot passa automaticamente para o próximo.

Após a identificação do tipo de *e-mail*, é extraído (da página do alerta) um relatório com extensão PDF com todas as informações do alerta (ID, *Source IP*, *Destination IP*, gráfico com informações durante o ataque, entre outros) que se representa na Figura A.9. O cliente é procurado na *collection*, tal como se apresenta na Figura A.8, e é enviado o *e-mail* com as informações necessárias (e com o ficheiro anexado). Sempre que é enviado um novo *e-mail* é colocada na *collection Collection to Merge* (apresentada na Figura 5.8) para adicionar os novos alertas tratados no *log*. A adição dos novos alertas tratados no *log* apresenta-se na Figura A.10 (comandos *Loop to Write Log* e *Append Log Collection*).

Para garantir que o ambiente da máquina fica nas mesmas condições do início do processo, é feito o *logout* no *Arbor* e é o IE é encerrado (esta etapa apresenta-se na Figura A.11).

Com a lógica do processo totalmente construída, todos os *objects* criados são passados para um único *process* conforme se apresenta na Figura 5.9.

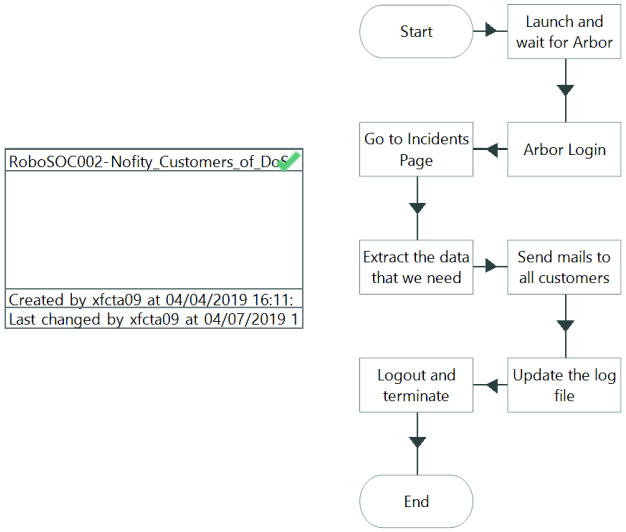


Figura 5.9: Processo do caso de uso *Notify Customers of DoS Attacks with Reports*

5.2.2 Segunda Iteração

Nesta iteração foi feito um *upgrade* da versão do *Blue Prism* que passou da versão 6.4.0 para a 6.5.0, sendo que nesta nova versão foi possível associar os *browsers Google Chrome* e *Mozilla Firefox* ao *Application Modeller*. O *browser* escolhido nesta iteração foi o *Google Chrome* e devido a essa mudança, foram criados novos *objects*. O *Process Studio* foi usado de forma mais exaustiva visto que foram criados vários *processes* para modelizar o caso de uso. Esta modularização aumenta a facilidade na implementação e torna os *objects* e os *processes* independentes.

A estrutura mantém-se a mesma, contudo é inserida à lógica a estrutura interna do *Blue Prism*: a *Queue*. Os novos alertas são guardados na *Queue* com o ID como chave primária e, para além deste, são guardados outros dados (tempo do alerta, cliente, *source IP*, entre outros). Quando um alerta não está atualizado na *Queue*, é feito o tratamento para a sua atualização. Quando um alerta já existe, mas os dados permanecem os mesmos tanto na *Queue* como no *Arbor*, o robot itera para o próximo alerta. A *Main Page* deste caso de uso, apresentada na Figura 5.10, é composta por *objects* e *processes*.

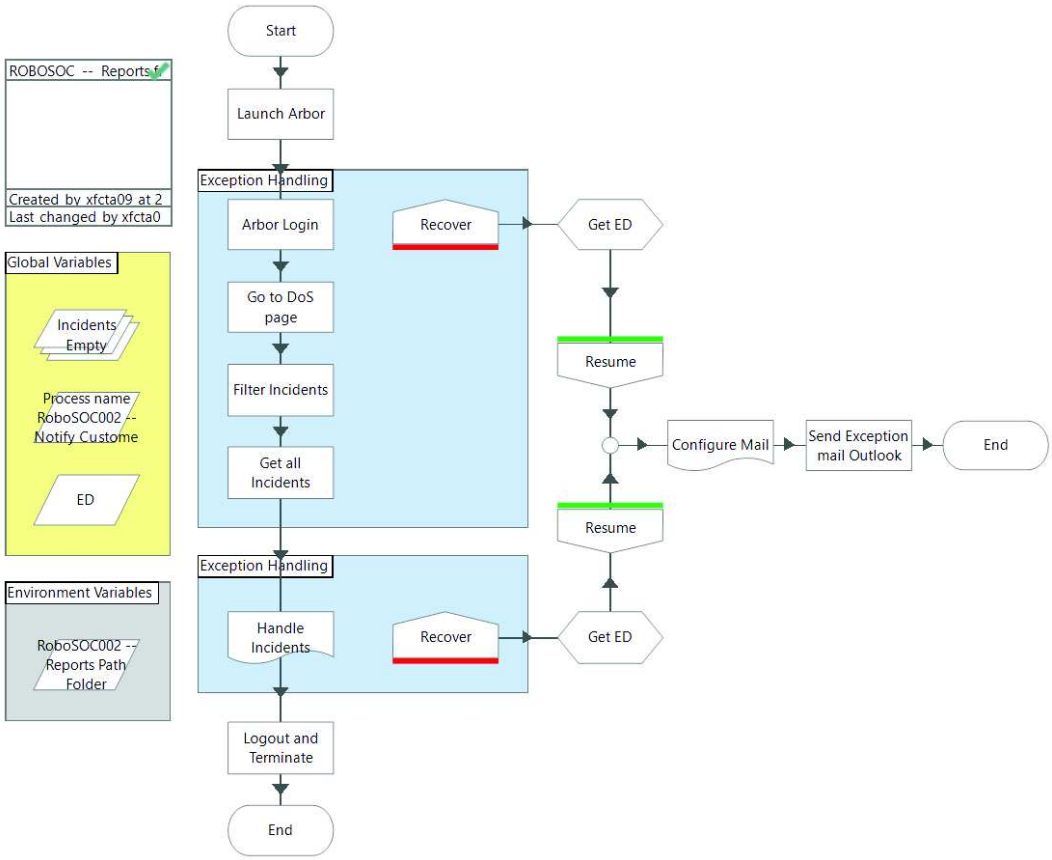


Figura 5.10: *Main Page* do tratamento dos alertas

O processo começa por lançar o *Arbor* a partir do *Google Chrome* (através de um *object*, ilustrado na Figura B.1). Quando o *browser* é iniciado, o problema dos certificados mantém-se e questiona o utilizador se faz questão prosseguir para a plataforma. No caso do *Chrome*, esta página não aparece todas as vezes que é iniciado, mas eventualmente retorna porque a entidade representante do certificado mantém-se desconhecida (Figura 5.10). Segue-se o *login* no *Arbor* (implementação ilustrada conforme a Figura B.2), onde são extraídas as credenciais guardadas no *Blue Prism* que são inseridas nos elementos compostos pelo *login* (barras de *username*, *password* e botão de *login*).

Quando termina o carregamento da página inicial do *Arbor*, o robot direciona-se para a página dos alertas de *DoS* (apresentado na Figura B.3) e filtra os alertas de modo a que apareçam 100 alertas de DoS numa única página. Após o aparecimento dos 100 alertas, o robot extrai todos os alertas e guarda-os numa *collection* chamada *Incidents* (Figura 5.10) que guarda todos os dados de cada alerta. O comportamento do *object* correspondente *Get all Incidents* ilustra-se na Figura B.5 onde são feitas várias manipulações sobre a extração para corrigir a apresentação de caracteres.

Com os alertas guardados, é feita a gestão dos mesmos (ilustrado na Figura 5.11) no subprocesso, como representado na *Main Page, Handle Incidents*. Neste subprocesso, a *collection* que tem os alertas é iterada e verifica se o alerta está na *Queue*, caso esteja, então pode ser necessário completá-lo (ou não). Caso contrário, tem que se verificar se o alerta já terminou ou não. Este processo tem apenas como objetivo alocar corretamente cada alerta para o tipo de *e-mail*.

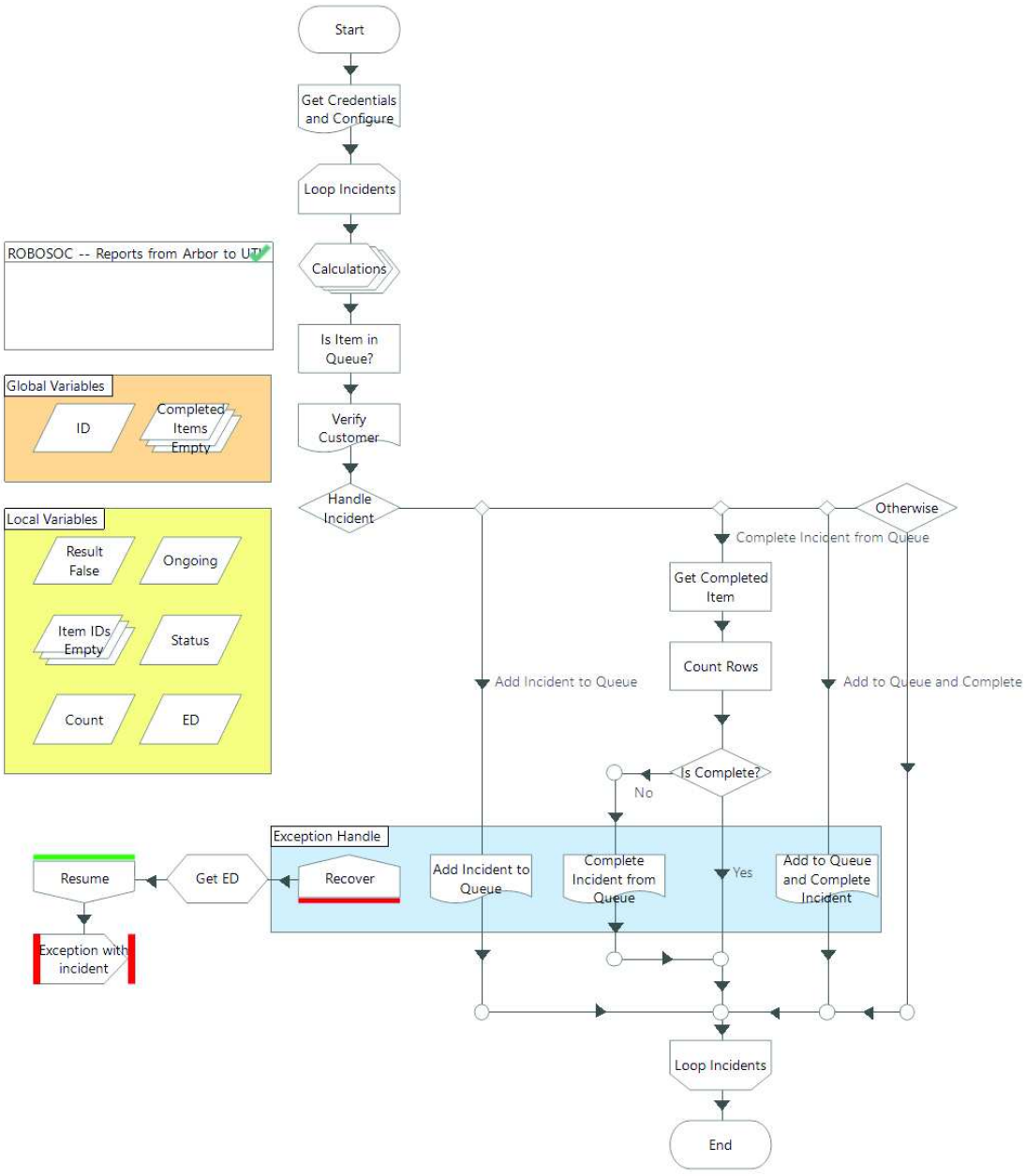


Figura 5.11: Processo que faz feita a gestão dos alertas

Existem quatro caminhos para cada alerta:

- O alerta nunca foi guardado pelo robot e o que faz a distinção do tipo de *e-mail* é o seu tempo que se representa por "dd MMM HH:mm yyyy – Ongoing" ou "dd MMM HH:mm yyyy – dd MMM HH:mm yyyy":

  1.1 **Caso exista o "Ongoing" no tempo do alerta**: significa que não terminou e por isso passa para o subprocesso *Add Incident to Queue* (apresentado na Figura 5.11 com a implementação ilustrada na Figura B.6). O alerta é inserido na *Queue* juntamente com os seus dados. Após a sua inserção, este fica *locked* (significa que está a ser processado). É feita a extração do relatório em formato PDF do alerta (apresentado na Figura B.7) e o robot verifica o *e-mail* correspondente ao cliente (Figura B.8). Por fim, é criado o *e-mail* de início do alerta com todas as suas informações e com o relatório anexado. Quando o *e-mail* for enviado, o alerta é *unlocked* para quando este terminar volte a ser processado;

  1.2 **Caso não exista "Ongoing" no tempo do alerta**: significa que iniciou e terminou antes do robot executar. O alerta passa para o subprocesso *Add to Queue and Complete Incident* (implementação apresentada na Figura B.9) e os seus dados são inseridos na *Queue* e fica marcado como *locked*. É feita a extração do relatório do alerta e é procurado o *e-mail* do respetivo cliente. Por fim, é criado um *e-mail* de início e fim do alerta com todos os dados e com o relatório em anexo. Após o envio do *e-mail*, o alerta dá-se como *Completed* (significa que não volta a ser processado).

- O alerta já foi guardado e foi enviado um *e-mail* de início, porém o alerta já terminou e o cliente ainda não foi notificado. Para notificar o término do alerta (apresentado na Figura B.10 e representado como *Complete Incident from Queue* na Figura 5.11) o estado do alerta é atualizado na *Queue* juntamente com os seus dados. De seguida é extraído o relatório. Após a extração, é criado o *e-mail* com os dados atualizados e com o novo relatório anexado. O cliente correspondente ao alerta é procurado e o *e-mail* é enviado. Após o envio do *e-mail*, o alerta é marcado como *Completed*;
- O alerta já foi guardado e totalmente tratado pelo robot.

Quando todos os alertas guardados estiverem tratados, é feito o *logout* no *Arbor* e o *browser* é fechado (apresentado na Figura B.11) deixando o ambiente da máquina nas mesmas condições que aquando o seu início.

Apesar dos alertas estarem num tratamento constante, o tempo de mitigação de cada um não é fixo. Ou seja, os ataques DoS possuem durações variantes. Sendo que a implementação apresentada só trata dos 100 alertas que aparecem na página do *Arbor*, podem existir outros alertas mais antigos que se mantêm pendentes na *Queue* (*i.e.*, já foram "vistos" pelo robot, mas não foram totalmente tratados). Estes alertas pendentes têm que ser tratados e para os completar, foi implementado um processo "extra" que verifica se cada alerta pendente na *Queue* já terminou ou não. A Figura 5.12 apresenta a estrutura principal do processo "extra" e o seu início é idêntico ao processo principal.

No processo da Figura 5.12, o subprocesso *Get Pending Items* (ilustrado na Figura B.12) obtém todos os elementos da *Queue* que tenham o estado pendente e insere-os numa *collection*. A *collection* é iterada e cada alerta é pesquisado, através do seu ID, diretamente no *Arbor*. Se no tempo do alerta estiver "Ongoing" significa que o ataque permanece ativo e o *Arbor* ainda está a mitigar. Caso contrário, o relatório é extraído e é criado um *e-mail* de término de um ataque para o respetivo cliente (com o relatório

anexado). Em suma, este processo tem como propósito garantir que todos os alertas passados pelo robot são totalmente tratados.

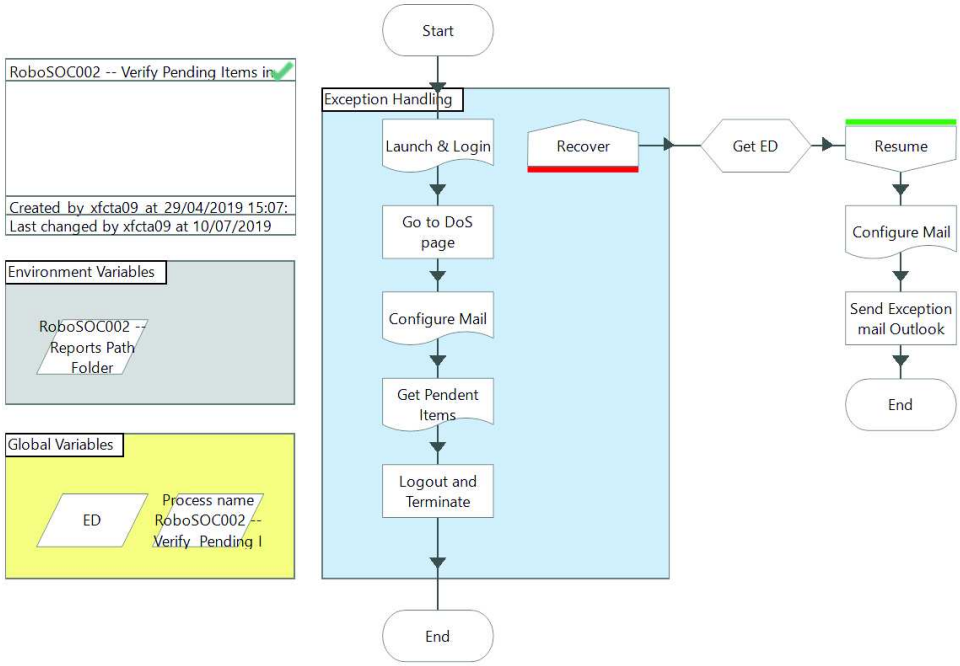


Figura 5.12: *Main Page* do processo *Verify Pendent DoS Attacks*

5.3 *Notify Customers of DoS Attacks with Screenshots*

Tal como é mencionado na Secção 4.4, este caso de uso tem a mesma finalidade que o da Secção 4.2. Acontece que devido à alteração de requisitos, o processo foi implementado numa única iteração e foi melhorado à medida das necessidades que iam sendo encontradas.

As aplicações usadas neste caso de uso foram *Outlook* e *Google Chrome* e o *browser* sempre que lançado, inicia-se como página inicial a página de *login* da plataforma *Arbor*. Para a implementação deste caso de uso, foram feitos três *objects* diferentes e um único *process* composto por vários subprocessos que utilizam os *objects* criados. Parte dos *objects* usados no *process* são do processo da Secção 5.2 e outros foram adicionados para a realização do caso de uso nas condições desejadas.

O processo foi dividido em duas partes: a primeira parte é composta pela extração dos *tickets* do sistema de *Request Tracker for Incident Response* (RTIR) e inserção dos novos *tickets* na *Queue*; e a segunda parte é a recolha dos elementos existentes na *Queue* e fazer o tratamento de cada um deles.

A implementação da parte correspondente à extração dos *tickets* e inseri-los na *Queue* ilustra-se na Figura 5.13. No processo da Secção 5.2 os alertas são extraídos diretamente da página do *Arbor* porque não existem restrições a nível da severidade, contudo neste caso de uso só serão tratados os alertas com o nível de severidade *High*. Como indicado anteriormente, na ocorrência de um alerta com este nível de severidade, é gerado um *ticket* (que fica guardado na base de dados). Para obter esses *tickets*, cria-se uma conexão entre o *Blue Prism* e a base de dados do RTIR e extrai-se todos os *tickets* gerados nos últimos sete dias. O subprocesso responsável pela conexão e obtenção dos *tickets* apresenta-se na Figura C.1.

A obtenção dos *tickets* é feita através da seguinte *query*:

```
select alert_id=master.dbo.PARSE_STRING2(Subject,'#',' ',1),*
from RTIR.dbo.Extract_RTIR_SOC_incidents_last_7_days
where Subject like 'Arbor Alert%';
```

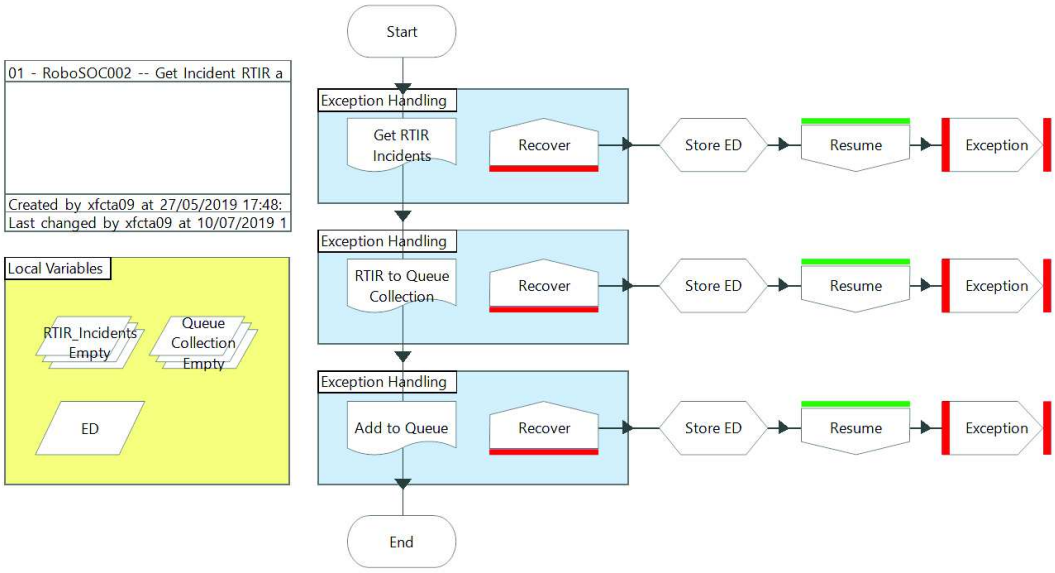


Figura 5.13: *Main Page* da parte *Get Incidents from RTIR* do caso de uso *Notify Customers of DoS Attacks with Screenshots*

Todos os *tickets* resultantes da *query* são transferidos para a *collection RTIR_Incidents*, apresentada na Figura 5.13. Todos os dados do *RTIR_Incidents collection* são transferidos para uma nova *collection* chamada *Queue Collection* que se apresenta na Figura C.2.

Após a transferência de *tickets* entre *collections*, é realizada a etapa da inserção dos novos *tickets* na *Queue*. A *Queue collection* é iterada e é verificada a existência do *ticket* na *Queue*: caso não exista, o *ticket* mantém-se na *collection*; caso contrário, a linha desse *ticket* é apagada da *collection* para evitar *tickets* repetidos na *Queue*, como é apresentado na Figura C.3. Todos os *tickets* possuem um de dois estados: *open* ou *resolved*. Esta etapa é realizada num único *process* mas dividida em:

- *Tickets* com estado *resolved*. Dentro do contexto do *Arbor* significa que o *ticket* foi aberto pelo SIEM e foi fechado pelo *Arbor* porque a mitigação terminou.;
- *Tickets* com estado *open*. Dentro do contexto do *Arbor* significa que o *ticket* foi somente aberto pelo SIEM, contudo ainda não houve nenhum evento que o fechasse (*e.g.*, o fim da sua mitigação). Relativamente ao processo para inserir os *tickets* com estado *open*, é igual aos *tickets* com estado *resolved*.

Depois da inserção dos *tickets* na *Queue*, o seu estado dentro da estrutura de dados do *Blue Prism* é pendente. Na *Queue*, os elementos inseridos podem ter quatro estados:

- ***Pendent***: significa que ainda não foi tratado e está em espera;
- ***Locked***: significa que o elemento está a ser tratado no momento. Para voltar a usufruir deste elemento, tem que se fazer *unlock* do mesmo para voltar ao estado pendente;
- ***Exception***: significa que o elemento foi processado, contudo envolveu uma exceção no seu tratamento. Quando isto acontece, é dada a possibilidade de criar um clone do elemento para ser tratado na próxima vez que o processo seja executado. Se não for criado um clone, o elemento não volta a ser tratado;

- ***Completed***: significa que o elemento foi processado com sucesso sem envolver qualquer problema no seu tratamento e não voltará a ser processado.

Sendo que todos os novos *tickets* estão com estado pendente, o processo passa para a segunda parte: iterar os elementos pendentes da *Queue* e processar um de cada vez. O começo deste processo é preparar o ambiente da máquina para que se possa processar os *tickets*, *i.e.*, iniciar o *browser* na página do *Arbor* e iniciar o *Outlook* para o envio dos *e-mails* de notificação aos clientes. A preparação do ambiente apresenta-se na Figura 5.14.

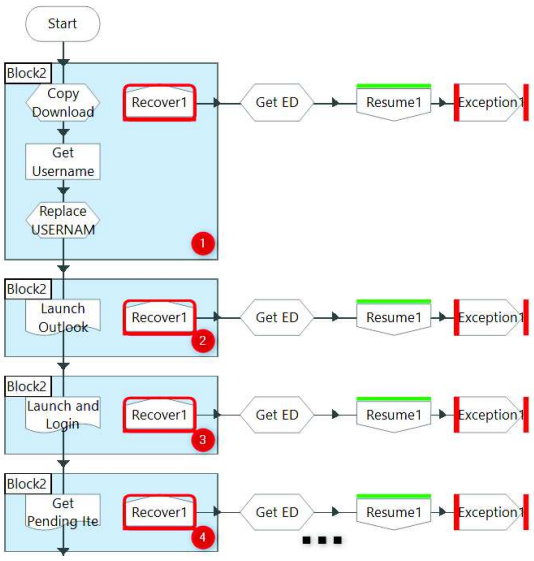


Figura 5.14: Preparação do ambiente da máquina

Antes de iniciar as aplicações necessárias para o tratamento dos *tickets*, é feita a alteração do *path* da pasta *Downloads* da máquina correspondente ao utilizador correto. Sempre que o processo for executado é preciso indicar qual o utilizador que faz o *login* na máquina, então, por questões de precaução, o *Blue Prism* obtém o *username* e faz o *replace* do *path*. Por exemplo, o *path* base é "C:\Users\username\Downloads" e quando a máquina tem sessão iniciada com o *user* SaraN, o *Blue Prism* altera o *path* para "C:\Users\SaraN\Downloads".

Os blocos 2 e 3 (da Figura 5.14) representam as aplicações a serem iniciadas e o bloco 4 é o subprocesso que recolhe os elementos pendentes na *Queue* e colocados numa *collection* (*output* do subprocesso) do processo para ser iterada, tal como se pode ver na Figura C.4. Para saber se é de facto necessário fazer o processo todo, são contabilizados quantos elementos estão pendentes na *Queue*. Se não existirem elementos para serem tratados, é passado diretamente para os subprocessos da Figura 5.15 que correspondem ao fecho das aplicações.

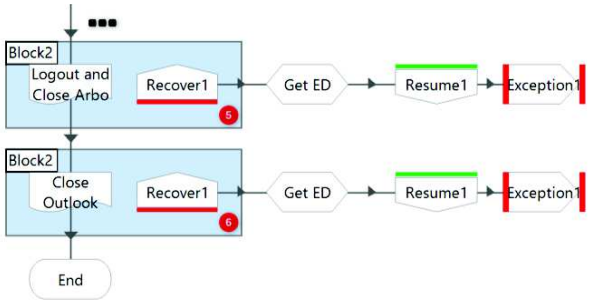


Figura 5.15: Etapa responsável pelo fecho das aplicações

Caso hajam elementos para tratar, é iniciado o *loop* na *Queue Collection* e no início do tratamento de um novo elemento, são feitas as verificações apresentadas na Figura 5.16. O passo *Is Stop Requested* verifica se a meio da execução do processo foi requerida a sua paragem. Isto impede que um elemento fique *locked* e não se possa tratá-lo novamente. Se a *Queue* estiver vazia, as aplicações são encerradas e dá-se o processo como terminado. Caso contrário, é verificado se o elemento está fechado e o *e-mail* de notificação foi enviado. Para o caso de estar fechado e o cliente notificado, é passado para o próximo elemento, e se não houverem mais elementos encerram-se as aplicações (Figura 5.15).

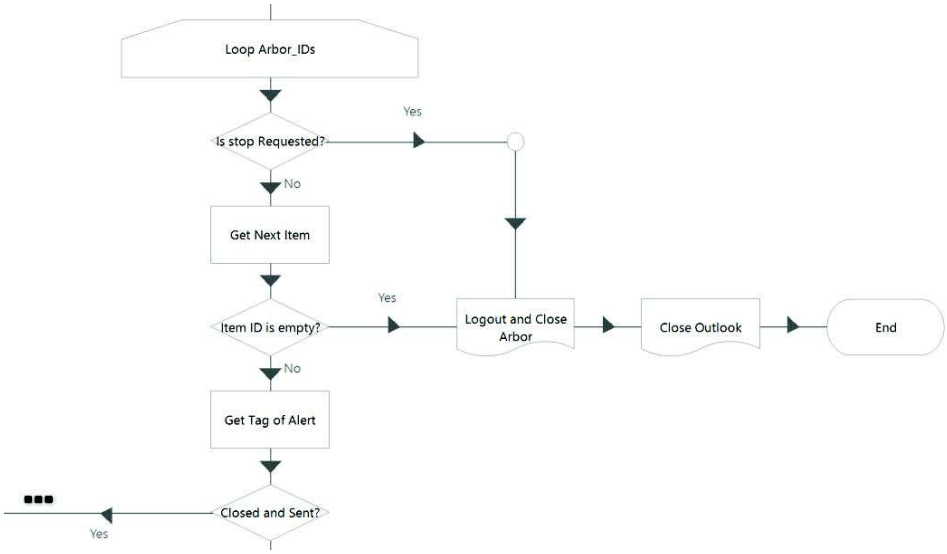


Figura 5.16: Verificações iniciais para o tratamento de um novo elemento

Nas condições opostas será realizado o processo de extração de informação na plataforma *Arbor* e a composição do *e-mail* de notificação ao respetivo cliente. O início deste processo de obtenção de informação apresenta-se na Figura 5.17.

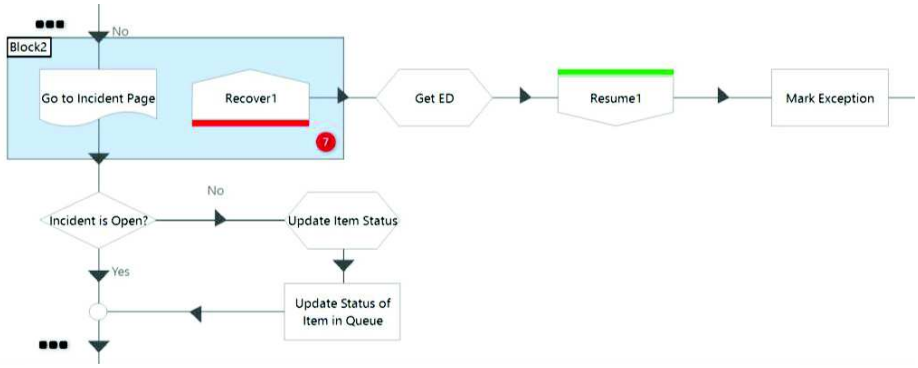


Figura 5.17: Início do processo de extração dos dados de um alerta

Para saber se existem atualizações no alerta, é criado um subprocesso ilustrado na Figura C.5, que faz a pesquisa do alerta através do seu ID na plataforma de mitigação (Figura C.6) e de seguida são extraídas as informações apresentadas no resultado. Para a extração da informação reutilizou-se o *object Get all Incidents* na Figura B.5. A primeira verificação que se faz é saber se o alerta mantém-se aberto como estava guardado na *Queue* e para o caso de ocorrerem alterações na informação, esta é atualizada para inserir futuramente na *Queue*. Caso se mantenha, o processo continua.

Como mencionado anteriormente, foram feitas verificações de se o alerta estava fechado e o cliente tinha sido notificado. Também são feitas verificações para alertas que ainda estejam em execução. Caso tenham sido notificados, é passado para o próximo alerta ou se não houver mais alertas para tratar, são fechadas as aplicações. Caso contrário, continua o processo de extração de informação tal como se apresenta na Figura 5.18.

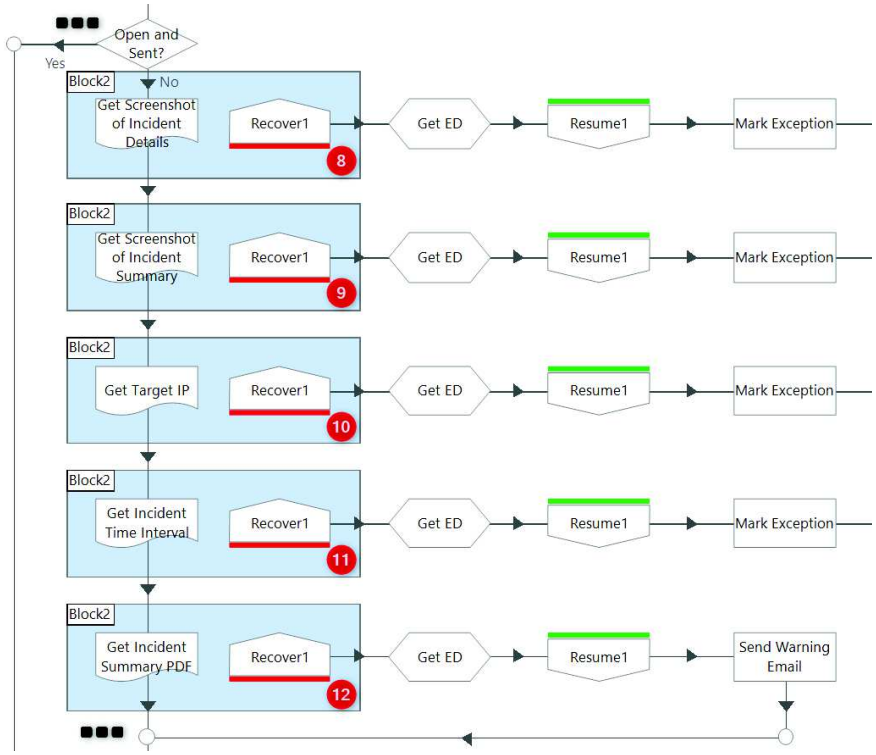


Figura 5.18: Obtenção dos *screenshots* e do relatório do sumário do alerta

A primeira extração a ser realizada situa-se no ponto oito da Figura 5.18 corresponde a um subprocesso onde é feita novamente a pesquisa do alerta através do seu ID (Figura C.6) e de seguida, é feito um *screenshot* semelhante ao da Figura 4.11 e é dado como *output* do *object*, representado na Figura C.7, para se guardar na *Main Page* do processo e inserir no *e-mail*.

De seguida, é feita a segunda extração de dados para a composição do *e-mail* que se representa nos pontos nove, dez e onze da Figura 5.18. No ponto nove, o robot redireciona-se para a página do sumário do alerta através de um URL (*Uniform Resource Locator*) com o ID do alerta no mesmo. Seguidamente é feito o *screenshot* semelhante à Figura 4.12 e para garantir que são sempre extraídos os conteúdos necessários (*i.e.*, não extrair conteúdo a mais ou haver falta do mesmo) são usados os *bounds* dos elementos e entre dois – ou mais – elementos conseguimos indicar a partir de que coordenada do ecrã começa e termina o *screenshot*. Por exemplo na Figura 5.19 é apresentada a identificação de um elemento "DoS Host Alert 430137" e as linhas que o enquadram. A implementação desta etapa ilustra-se na Figura C.8.

Figura 5.19: Exemplo de um elemento identificado pelo *Blue Prism*

Os blocos dez e onze retiram o *source* IP e os tempos do ataque (*start time* e *end time*) para escrever no *e-mail* de notificação. Por fim, para terminar a obtenção de dados relacionados com os detalhes do alerta, é extraído o ficheiro em formato PDF (ilustrado na Figura C.9) e é guardado numa pasta de rede para a eventualidade de ser necessário analisar o alerta.

Após estes cinco passos, resta a extração dos dados relacionados com a mitigação do ataque. A etapa de extração dos dados da mitigação representa-se na Figura 5.20.

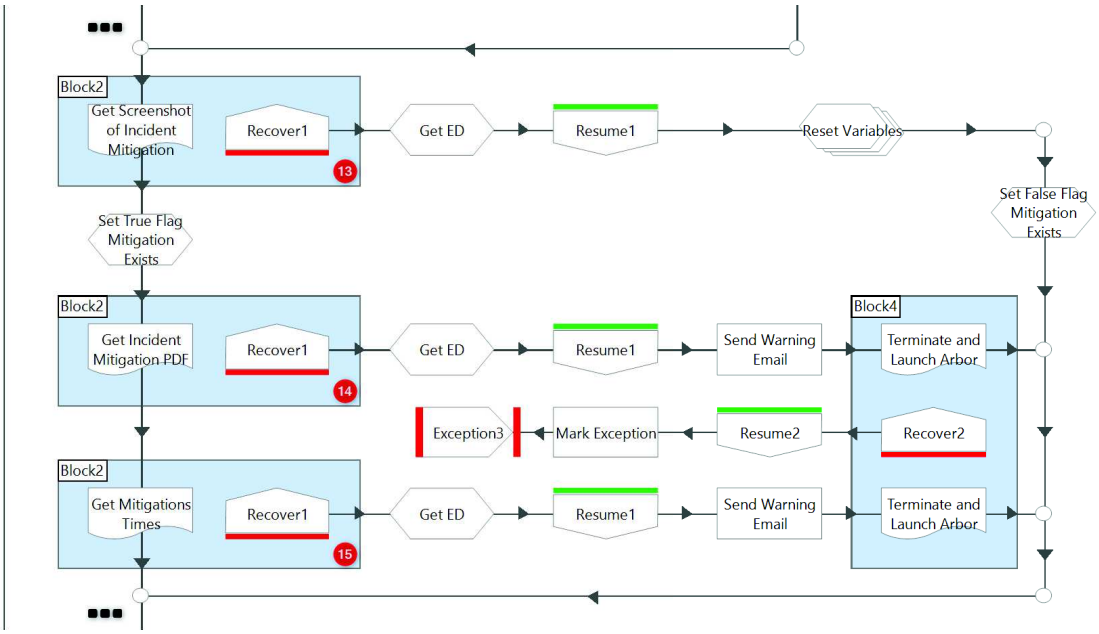


Figura 5.20: Etapa de extração das informações da mitigação

Acontece que a fase de mitigação não decorre em todos os alertas. Isto deve-se ao facto de – por vezes – existirem duplicações de alertas (ou seja, alertas diferentes, com IDs distintos representando o mesmo ataque) e devido a este contratempo, somente um deles tem mitigação. No ponto 13 da Figura 5.20, para evitar que o processo termine com um erro, é verificado se existe pelo menos um resultado após a pesquisa do ID do alerta (Figura C.10, parte destacada). Para o caso de não existir, o processo continua.

Na condição de existir mitigação, o robot mantém-se no ponto 13 e direciona-se para a página de mitigação do alerta e faz o *screenshot* (semelhante à Figura 4.13 e a sua obtenção representa-se na Figura C.10). O *screenshot* é extraído em quatro partes como é apresentado na Figura 5.21. Entre todas as partes extraídas, somente a terceira não será usada para a constituição do *e-mail* porque não contém informação relevante para o cliente. Cada parte tem dois elementos *web* e são extraídos os *bounds* dos mesmos que servem de referência para saber em que pontos é feito o recorte (elementos identificados em retângulos destacados).

Foi usada esta abordagem de recorte devido a um problema na máquina de produção. Os *screenshots* no *Blue Prism* são dependentes da resolução do ecrã da máquina que está em execução. E a resolução da máquina de produção é de 1021x592 (resolução *default* da *driver*), o que impedia a obtenção de toda a imagem em um *screenshot*. Por estes motivos, teve que se repartir a figura em partes e todas elas são "coladas" no *e-mail* compondo uma única figura.

De seguida, o ponto 14 é extraído o relatório de mitigação em formato PDF e são extraídos para variáveis de texto os tempos de começo e término da mitigação do alerta (representado no ponto 15 da Figura 5.20 e na Figura C.11).

Figura 5.21:Exemplo do recorte do *screenshot* da mitigação

Quando a extração dos dois relatórios é realizada, ambos são transferidos para a pasta nomeada como *default* do *Google Chrome* que é a pasta *Downloads*. Contudo, cada relatório deverá ser transferido para a pasta correspondente. As transferências dos relatórios apresentam-se na Figura 5.22.

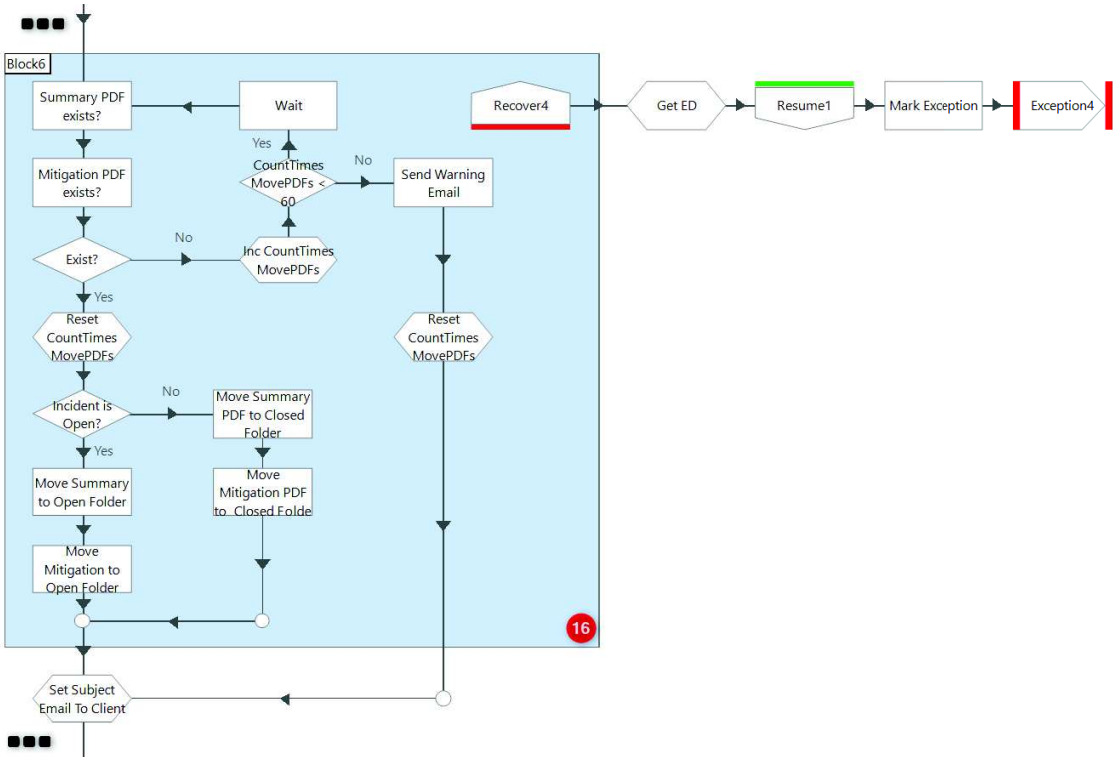


Figura 5.22: Etapa responsável pela transferência dos relatórios do alerta

Visto que os relatórios têm tamanhos variáveis, a duração de transferência é incerta. Por vezes a rede poderá estar fraca (ou seja, em vez de um ficheiro levar 10 segundos a ser extraído, leva mais que o dobro do tempo) e retardando a transferência dos mesmos. Visto que o robot faz os passos em milésimos de segundos, verifica-se durante um minuto a existência dos relatórios na pasta *Downloads* para saber se pode transferir os ficheiros nas pastas de destino. Se os relatórios forem detetados antes do tempo terminar, estes são movidos. Se não, é enviado um *e-mail* a notificar a equipa de que houve um problema relacionado com os relatórios. Tal como mencionado, nem sempre há mitigação, logo nem sempre há relatório de mitigação. Porém, apesar de não existir relatório de mitigação, o passo correspondente à transferência do mesmo não devolve erro.

Por fim, é criado o *subject* do *e-mail* de notificação que tem a estrutura seguinte:

```
[csirt.telecom.pt #<ID do ticket>] Ataque DDoS com destino a <IP de destino>
```

O ID do *ticket* corresponde ao alerta que está em tratamento e no fim é inserido o IP de destino, *i.e.*, o IP da máquina que está a sofrer o ataque. O ID do *ticket* obtém-se nos dados do alerta e a extração do IP de destino representa-se no bloco 10 da Figura 5.18.

Por fim, para terminar o tratamento de um alerta é criado o *e-mail* de notificação para o respetivo cliente e o ambiente é colocado nas condições para o tratamento do próximo alerta (Figura 5.23). No bloco 17 é feita a conversão das datas apresentadas no *Arbor* para o estilo *standard date and time format* do *International Organization for Standardization* (ISO) [19]. As conversões das datas apresentam-se na tabelas D.1 e D.2 e estas serão guardadas para a sua inserção no *e-mail*.

De seguida, no bloco 18, é feita a estruturação do *e-mail* de notificação para o cliente. Contudo, a abordagem usada para a sua criação foi diferente porque não é possível inserir as imagens diretamente no *e-mail* através de uma função VBO (Secção 3.3.2). Por este motivo, foi preciso criar um novo *object* com a aplicação *Outlook* associada ao mesmo. A Figura C.12 apresenta o início da criação do *e-mail* e a primeira parte destacada representa o robot a abrir uma janela de um *e-mail*. Depois são inseridos os dados iniciais – *To*, o *Carbon Copy* (CC) e o *subject*. Por último, é verificada a hora e conforme a mesma é inserido o cumprimento.

De seguida, dentro da composição do *e-mail* é inserido um texto inicial a especificar ao cliente a ocorrência do ataque de DoS e qual a máquina que está a sofrer do mesmo. O texto inserido é o seguinte:

```
No âmbito do serviço SOC contratado, informamos que foi detetado um
ataque de navegação de serviço com origem distribuído e com destino ao
endereço IP <IP de destino>, entre as <hora de início do ataque> e as
<hora de término do ataque>, tendo a seguinte timeline:
```

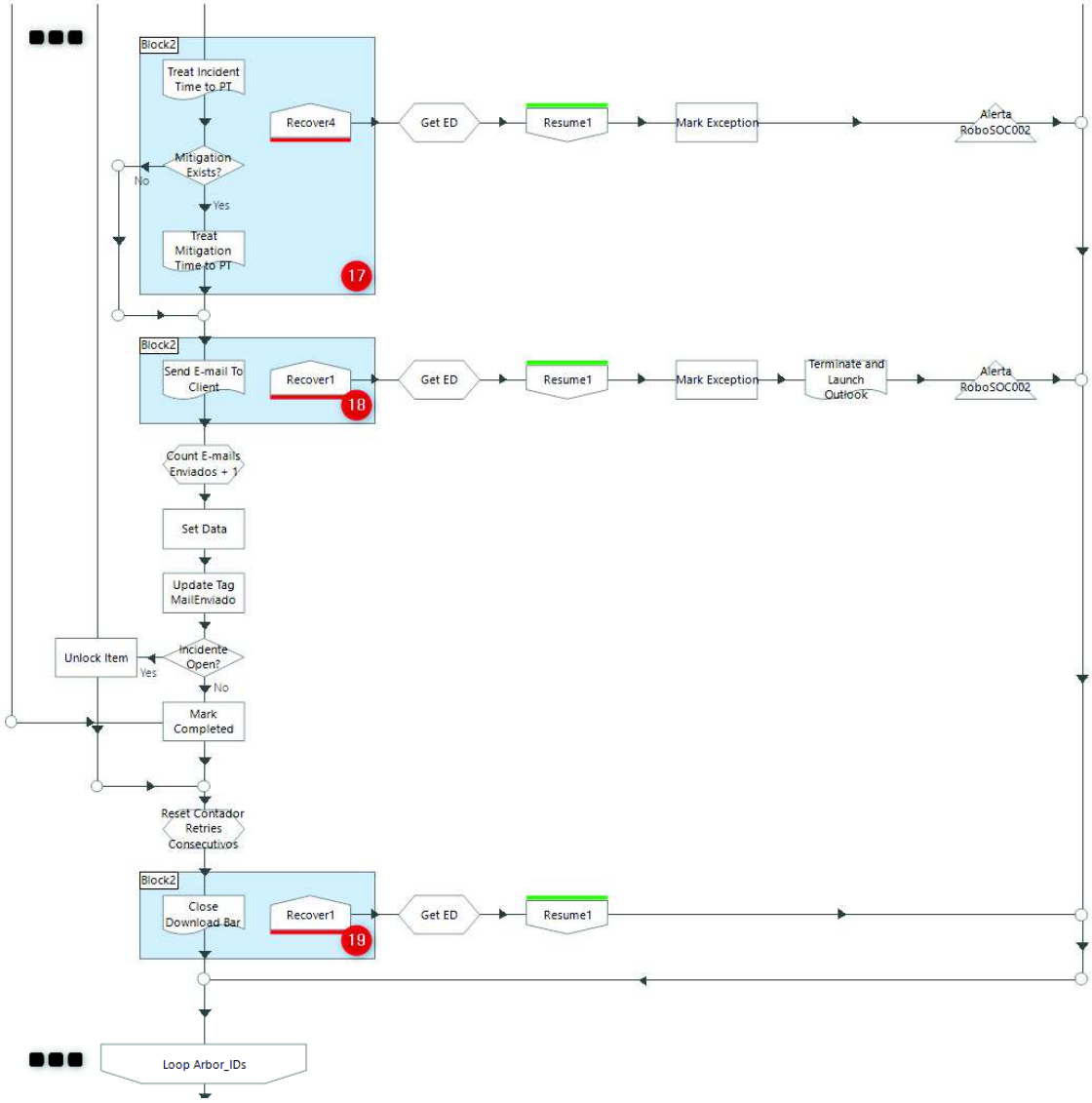


Figura 5.23: Criação do *e-mail* de notificação e parte final do tratamento do alerta

Após a inserção do texto inicial, é criada uma tabela que resume os tempos de início e término do ataque e os tempos da mitigação. Esta tabela (enviada ao cliente) tem o propósito de simplificar a informação do alerta e facilitar a análise do cliente. Quando a tabela estiver criada e preenchida, são inseridos os *screenshots* feitos anteriormente e todos eles são "colados" na mesma ordem que foram extraídos. É inserida a assinatura da equipa SOC (Figuras C.13 e C.14) e envia-se o *e-mail*. Após o envio, são atualizadas as informações do alerta na *Queue* e verifica-se novamente se o alerta está aberto. Se o alerta estiver aberto: é feito o *unlock* do elemento na *Queue* para quando o alerta terminar, fazer o tratamento final do mesmo. Se o alerta já estiver terminado: muda-se o estado do elemento na *Queue* para *Completed* e este alerta nunca mais será tratado.

O bloco 19 representa uma última interação com o *Chrome* e limita-se ao fecho da barra de *downloads* na própria aplicação. Esta interação é crucial porque se esta barra estiver presente quando forem feitos os *screenshots*, aparecerá sobre informação relevante para o *e-mail*. Ou seja, quando for feito o tratamento do próximo elemento, a plataforma *Arbor* estará nas mesmas condições que quando o processo iniciou.

Quando todos os elementos pendentes na *Queue* estiverem tratados e atualizados, é feito o *logout* da plataforma *Arbor* e fecha a aplicação *Google Chrome* consecutivamente. Por fim, é fechada a aplicação do *Outlook*. Estes passos representam-se na Figura 5.15.

Capítulo 6
Resultados

Neste capítulo são apresentadas métricas para avaliação da robotização dos casos de uso implementados. Juntamente, são apresentados os resultados por cada caso de uso e quais os ganhos respetivos.

6.1 Métricas para avaliação da robotização

A avaliação dos casos de uso baseou-se em três métricas. Estas foram usadas para avaliar os ganhos obtidos após a inserção dos robots no ambiente SOC. As métricas foram as seguintes:

- **Tempo de execução do processo**: diferença entre o instante de conclusão e o instante de início do processo;
- **Percentagem da taxa de erro**: é a fração entre o número de vezes que falhou na execução do processo e o número total de vezes que foi executado;
- **Disponibilidade**: percentagem do tempo disponível que o ator tem para executar o processo.

Os processos postos em produção foram colocados em datas distintas e todos os resultados serão apresentados com base no tempo que estiveram em produção (ou desenvolvimento) e o número de execuções feitas durante esse tempo.

Apesar da abordagem correta ser colocar o processo em ambiente produção quando estiver totalmente implementado (*i.e.*, com o levantamento de exceções incluído), é preciso testar várias vezes o processo dentro do ambiente de produção para garantir que, no final, execute sem erros. Por este motivo, dentro do período de observação dos casos de uso, são contabilizadas as execuções dos testes e das execuções agendadas no *Blue Prism*.

6.2 *Daily Reports from ArcSight to Tableau*

O caso de uso *Daily Reports from ArcSight to Tableau* é executado diariamente e corresponde à transferência dos anexos de dois *e-mails* vindos do SIEM *Micro Focus ArcSight* para uma pasta de rede onde serão processados na plataforma *Tableau* para fins analíticos.

A disponibilidade é um ponto crucial para a boa execução deste processo. Isto deve-se a dois pontos:

- O processo de transferência tem que ser feito antes das 11h30 da manhã (todos os dias) porque está programado no *Tableau* uma regra para buscar os ficheiros na pasta de rede para processar;
- Se o processo de transferência for feito após essa hora, o *Tableau* apenas irá buscar os anexos no dia seguinte, ou seja, em vez de dois anexos processa quatro (os do dia anterior e os atuais) e a plataforma não reconhece que os anexos são de dias diferentes então agrega todos os resultados no mesmo gráfico o que, consequentemente, apresenta resultados incorretos.

Os analistas SOC fazem atualmente uma distribuição deste processo para não sobrecarregar constantemente a mesma pessoa. Contudo, mesmo havendo um responsável, pode haver o risco do analista não transferir os anexos por esquecimento, ou por falta de recursos (por exemplo, uma máquina com uma VPN para ligar à rede da MEO caso esteja fora do trabalho ou falta de rede no trabalho). Já o robot está disponível todos os dias e executa o processo a tempo. Este processo foi agendado para ser executado às 09h15 da manhã, tal como se pode ver na Figura 6.1.

O ponto 1 apresenta o nome do agendamento. Os pontos 2 e 3 indicam em propriedades como: qual o primeiro processo quando é iniciada a execução e a que horas é executado, respetivamente.

Este processo foi colocado no ambiente de produção a partir do dia 15 de Maio de 2019 e o intervalo de observação durou até ao dia 25 de Julho de 2019. Dentro deste intervalo de tempo, o número total de execuções feitas deste processo foram 121 sendo que 10 terminaram em exceção.

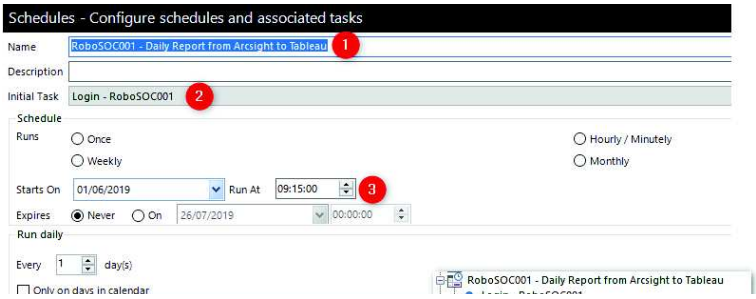


Figura 6.1: Composição e agendamento do processo *Daily Reports from ArcSight to Tableau*

Geralmente o tempo de execução do processo realizado pelos analistas baseia-se na procura dos *e-mails* do *ArcSight* e copiar os anexos para a pasta destino e tem uma duração entre 60 a 90 segundos. Para além disso, existe a possibilidade de errar na transferência (por exemplo, mover os anexos na pasta errada). No caso do robot, o tratamento de cada anexo leva entre 1 a 3 segundos, tal como se pode ver na Figura 6.2.

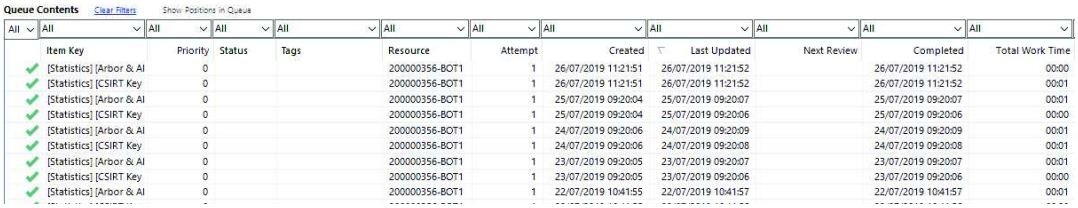


Figura 6.2: Representação dos tempos de tratamento de cada anexo na *Queue*

Como se pode ver, para além do tempo reduzido do tratamento dos anexos, todos foram tratados com sucesso sem resultar no lançamento de uma exceção. Deste modo, pode-se concluir que durante o período de observação a fiabilidade foi superior a 90% em 121 execuções. Isto permite que o analista tenha disponibilidade para realizar processos mais importantes do SOC.

Enquanto um analista SOC demora entre 60 a 90 segundos a concretizar este processo, o robot demora entre 3 a 6 minutos. Este aumento deve-se ao tempo de espera entre os passos do processo (*i.e.*, entre o *login*, o *Run Process* e o *logout* apresentados na Figura 6.1) porque após o *login* configurou-se o robot para esperar dois minutos para a máquina estar preparada para executar o processo de transferência. Para além disso, o robot tem uma regra que espera no máximo cinco minutos para ter a *Inbox* do *Outlook* atualizada. Devido aos motivos apresentados, o tempo de execução do processo aumenta.

O resumo da avaliação da execução do processo entre os analistas SOC e o robot apresenta-se na Tabela 7:

Critérios	Analista SOC	Robot
Tempo de execução	1 – 1.5 Minutos	3 – 6 Minutos
Percentagem de taxa de erro	*	≈ 8.3 %
Disponibilidade	?	100%

? – Possibilidade do analista estar doente ou de férias ou ser feriado

* – Não é possível calcular

Tabela 7: Valores das métricas para o analista e o robot do caso de uso *Daily Reports from ArcSight to Tableau*

Como a Tabela 7 ilustra, todas as execuções ocorreram a horas e por isso conclui-se que o robot está totalmente disponível para a realização do processo. Face à disponibilidade do analista, não é dado um valor porque há vários parâmetros que devem ser contabilizados tais como: se o analista está doente ou não, se está de férias, se tem os recursos necessários, entre outros. Relativamente à percentagem de taxa de erro um parâmetro que se deve considerar é o cansaço, ou seja, à medida que o analista trabalha no mesmo processo, a longo prazo, é possível a taxa de erro aumentar.

6.3 *Notify Customers of DoS Attacks with Reports*

Este caso de uso é realizado diariamente e corresponde à extração dos alertas através do *browser* sem restrições ao nível de severidade do alerta.

Este caso de uso foi agendado para ser feito de dez em dez minutos por dois motivos:

- Primeiro não havendo restrições ao nível de severidade, há a consequência do volume de alertas a tratar ser maior;
- Há a ambição de notificar o cliente o mais rápido possível.

Este processo não foi posto em produção devido às alterações de requisitos como apresentado na Secção 4.4. Devido ao contratempo, o processo foi simulado no ambiente de desenvolvimento para apresentar alguns resultados e por consequência, não foi realizado o agendamento no ambiente de produção.

Para simular, criou-se um agendamento no ambiente de desenvolvimento, como se apresenta na Figura 6.3. Este agendamento só tem o processo da extração dos relatórios e notificação dos clientes pois foi executado nas máquinas de desenvolvimento. Por este motivo não foi preciso adicionar o *login* e o *logout*.

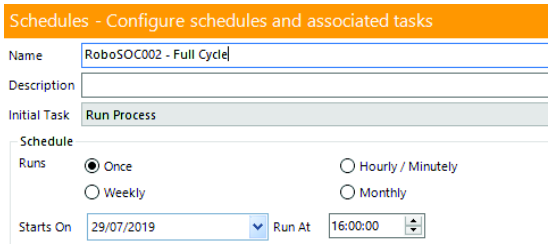


Figura 6.3:Composição e agendamento no ambiente de desenvolvimento do processo *Notify Customers of DoS Attacks with Reports*

As métricas não foram calculadas visto que o processo não foi colocado em produção. Contudo, foi executado 34 vezes desde do dia 4 de Abril de 2019 onde 6 execuções terminaram em exceção. Estas execuções foram realizadas para poder calcular uma média do tempo de execução como se apresenta a Figura 6.4.

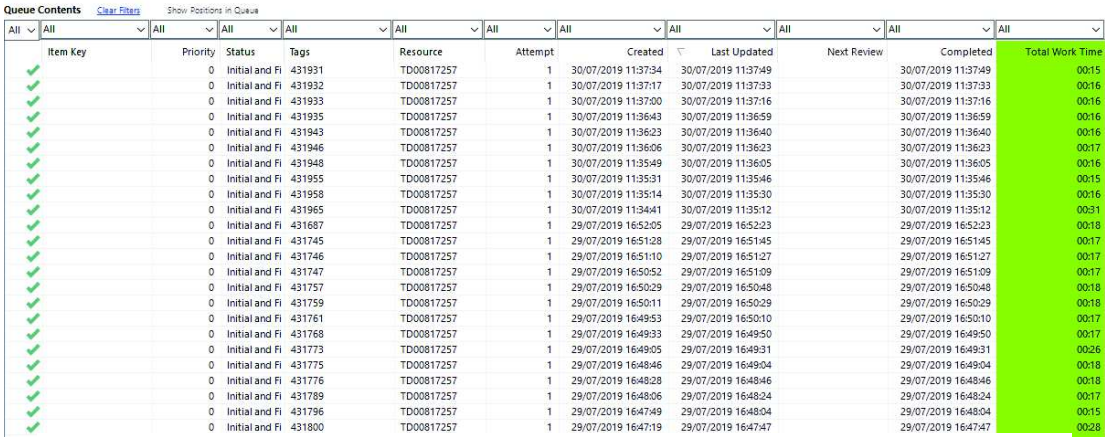
Environment Start Selected Sessions Stop Selected Sessions Show Session Variables

ID	Process	Resource	User	Status	Start Time	End Time
820	ROBOSOC -- Reports from Arbor t...	TD00817257	xfcta09	Completed	30/07/2019 11:34:15	30/07/2019 11:38:27
819	ROBOSOC -- Reports from Arbor t...	TD00817257	xfcta09	Completed	29/07/2019 16:46:54	29/07/2019 16:52:47
818	ROBOSOC -- Reports from Arbor t...	TD00817257	xfcta09	Completed	29/07/2019 16:19:43	29/07/2019 16:25:10
817	ROBOSOC -- Reports from Arbor t...	TD00817257	xfcta09	Completed	29/07/2019 16:04:22	29/07/2019 16:10:02
454	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	20/05/2019 17:07:10	20/05/2019 17:07:34
292	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	07/05/2019 09:39:54	07/05/2019 10:02:43
291	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	07/05/2019 09:39:04	07/05/2019 09:39:12
290	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	07/05/2019 09:36:37	07/05/2019 09:36:44
289	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	07/05/2019 09:36:10	07/05/2019 09:36:22
286	ROBOSOC -- Reports from Arbor t...	TD00817257	xfcta09	Completed	06/05/2019 11:09:53	06/05/2019 11:11:48
285	ROBOSOC -- Reports from Arbor t...	TD00817257	xfcta09	Completed	06/05/2019 11:08:53	06/05/2019 11:09:18
281	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	06/05/2019 10:34:30	06/05/2019 10:57:11
280	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	03/05/2019 12:21:32	03/05/2019 12:24:29
277	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	30/04/2019 18:18:00	30/04/2019 18:20:01
276	ROBOSOC -- Reports from Arbor t...	TD70001145	xfcta09	Completed	30/04/2019 18:17:19	30/04/2019 18:17:26

Figura 6.4: Exemplos de execuções do processo

Neste processo, a maior parte do tempo despendido compõe-se na extração dos relatórios dos alertas e no envio do *e-mail* de notificação de cada um deles. Como ilustrado na Figura 6.4, verifica-se que o tempo de execução do processo é variável. Esta variância temporal deve-se ao facto de que o número de alertas novos para tratar ser inconstante (ou seja, no mesmo espaço de tempo pode-se ter tanto dois como 20 alertas novos para tratar).

Os analistas SOC para tratarem um alerta demoram em média 5 minutos desde o momento que pesquisam o alerta no *Arbor* até ao momento que enviam o *e-mail* de notificação ao cliente. Com todas as execuções feitas, cada alerta consome entre 15 a 35 segundos (ilustrado na Figura 6.5), o que garante um aumento de 1100% no desempenho do processo.



Queue Contents Clear Filters Show Positions in Queue

Item Key	Priority	Status	Tags	Resource	Attempt	Created	Last Updated	Next Review	Completed	Total Work Time
	0	Initial and Fi	431931	TD00817257	1	30/07/2019 11:37:34	30/07/2019 11:37:49		30/07/2019 11:37:49	00:15
	0	Initial and Fi	431932	TD00817257	1	30/07/2019 11:37:17	30/07/2019 11:37:33		30/07/2019 11:37:33	00:16
	0	Initial and Fi	431933	TD00817257	1	30/07/2019 11:37:00	30/07/2019 11:37:16		30/07/2019 11:37:16	00:16
	0	Initial and Fi	431935	TD00817257	1	30/07/2019 11:36:43	30/07/2019 11:36:59		30/07/2019 11:36:59	00:16
	0	Initial and Fi	431943	TD00817257	1	30/07/2019 11:36:23	30/07/2019 11:36:40		30/07/2019 11:36:40	00:16
	0	Initial and Fi	431946	TD00817257	1	30/07/2019 11:36:06	30/07/2019 11:36:23		30/07/2019 11:36:23	00:17
	0	Initial and Fi	431948	TD00817257	1	30/07/2019 11:35:49	30/07/2019 11:36:05		30/07/2019 11:36:05	00:16
	0	Initial and Fi	431955	TD00817257	1	30/07/2019 11:35:31	30/07/2019 11:35:46		30/07/2019 11:35:46	00:15
	0	Initial and Fi	431958	TD00817257	1	30/07/2019 11:35:14	30/07/2019 11:35:30		30/07/2019 11:35:30	00:16
	0	Initial and Fi	431965	TD00817257	1	30/07/2019 11:34:41	30/07/2019 11:35:12		30/07/2019 11:35:12	00:31
	0	Initial and Fi	431687	TD00817257	1	29/07/2019 16:52:05	29/07/2019 16:52:23		29/07/2019 16:52:23	00:18
	0	Initial and Fi	431745	TD00817257	1	29/07/2019 16:51:28	29/07/2019 16:51:45		29/07/2019 16:51:45	00:17
	0	Initial and Fi	431746	TD00817257	1	29/07/2019 16:51:10	29/07/2019 16:51:27		29/07/2019 16:51:27	00:17
	0	Initial and Fi	431747	TD00817257	1	29/07/2019 16:50:52	29/07/2019 16:51:09		29/07/2019 16:51:09	00:17
	0	Initial and Fi	431757	TD00817257	1	29/07/2019 16:50:29	29/07/2019 16:50:48		29/07/2019 16:50:48	00:18
	0	Initial and Fi	431759	TD00817257	1	29/07/2019 16:50:11	29/07/2019 16:50:29		29/07/2019 16:50:29	00:18
	0	Initial and Fi	431761	TD00817257	1	29/07/2019 16:49:53	29/07/2019 16:50:10		29/07/2019 16:50:10	00:17
	0	Initial and Fi	431768	TD00817257	1	29/07/2019 16:49:33	29/07/2019 16:49:50		29/07/2019 16:49:50	00:17
	0	Initial and Fi	431773	TD00817257	1	29/07/2019 16:49:05	29/07/2019 16:49:31		29/07/2019 16:49:31	00:26
	0	Initial and Fi	431775	TD00817257	1	29/07/2019 16:48:46	29/07/2019 16:49:04		29/07/2019 16:49:04	00:18
	0	Initial and Fi	431776	TD00817257	1	29/07/2019 16:48:28	29/07/2019 16:48:46		29/07/2019 16:48:46	00:18
	0	Initial and Fi	431789	TD00817257	1	29/07/2019 16:48:06	29/07/2019 16:48:24		29/07/2019 16:48:24	00:17
	0	Initial and Fi	431796	TD00817257	1	29/07/2019 16:47:49	29/07/2019 16:48:04		29/07/2019 16:48:04	00:15
	0	Initial and Fi	431800	TD00817257	1	29/07/2019 16:47:19	29/07/2019 16:47:47		29/07/2019 16:47:47	00:28

Figura 6.5: Representação dos tempos de tratamento de cada alerta na *Queue*

No caso dos analistas existem várias particularidades que podem conduzir à realização de erros. O objetivo principal do caso de uso é notificar o cliente o mais rápido possível e para atingi-lo, o processo teria que usar uma *Event Driven Arquitecture* (EDA) [39].

Tal como já foi referido, o processo nunca foi colocado em produção porque foi desenvolvido outro processo semelhante. Não existindo dados reais para calcular as métricas, não são feitas avaliações do processo.

6.4 *Notify Customers of DoS Attacks with Screenshots*

O caso de uso *Notify Customers of DoS Attacks with Screenshots* é executado diariamente e fundamenta-se em notificar os clientes dos alertas de *DoS* através do correio eletrónico (onde a extração da informação dos mesmos constitui-se por *screenshots*) e foi posto em ambiente de produção.

Neste caso estamos a considerar alertas que tenham nível de severidade *High*, por esse motivo o número de alertas a serem tratados diminui face ao processo da Secção 6.3. Para além disso, a extração dos alertas é mais detalhada e a elaboração do *e-mail* também. Deste modo, sabe-se que cada alerta tem um tempo de tratamento superior face ao tratamento feito no caso de uso *Notify Customers of DoS Attacks with Reports*.

Para evitar grandes intervalos entre a atualização dos alertas e a notificação dos mesmos, o processo foi agendado para ser executado de 10 em 10 minutos tal como se apresenta na Figura 6.6.

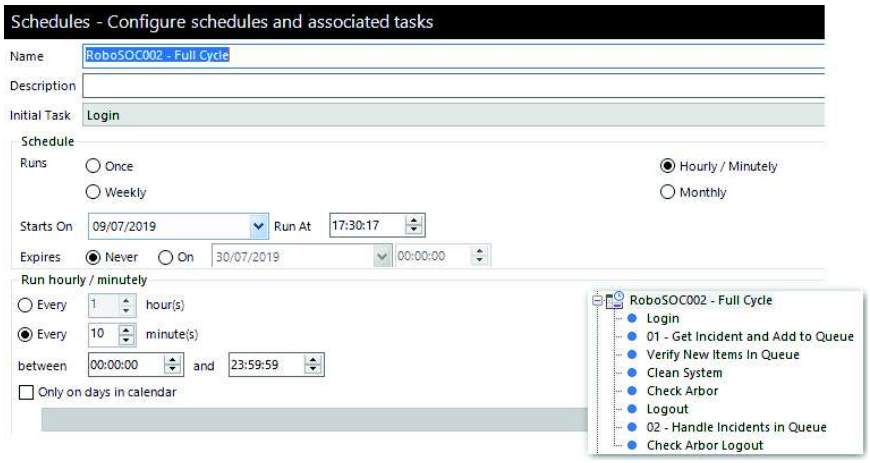


Figura 6.6: Composição e agendamento do processo *Notify Customers of DoS Attacks with Screenshots*

Em algumas execuções do processo, são encontradas adversidades relacionadas com a identificação dos elementos no *login* da plataforma *Arbor*. Para tentar solucionar o problema, foram aplicadas algumas abordagens da página de Suporte do *Blue Prism* e da equipa de suporte (que foi diretamente contactada), mas qualquer uma destas não resultou qualquer efeito. Deste modo, visto que era relevante colocar o processo em execução e como este problema não acontecia em todas as execuções foi tomada a decisão de primeiro testar o *login* do *Arbor*. Na situação dos elementos serem identificados, o processo é executado na sua totalidade. Caso contrário, faz-se *logout* e tenta-se na próxima execução.

Sendo que as execuções são feitas em intervalos de tempo curtos, por vezes, não há alertas novos a tratar. Na Figura 6.7 apresentam-se algumas execuções do processo no ambiente de produção onde somente foram selecionadas as partes principais do mesmo. Na maior parte dos casos, a parte *00 – Verify New Items* dá-se como *Terminated* porque não existem alertas novos, por isso, é feito o *logout* na máquina. Mas nas zonas destacadas verificam-se duas execuções consecutivas onde foram tratados alertas novos.

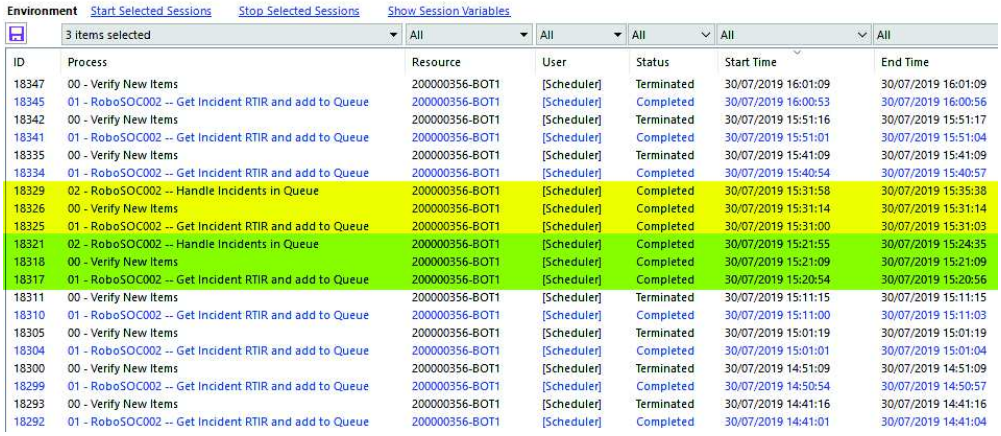
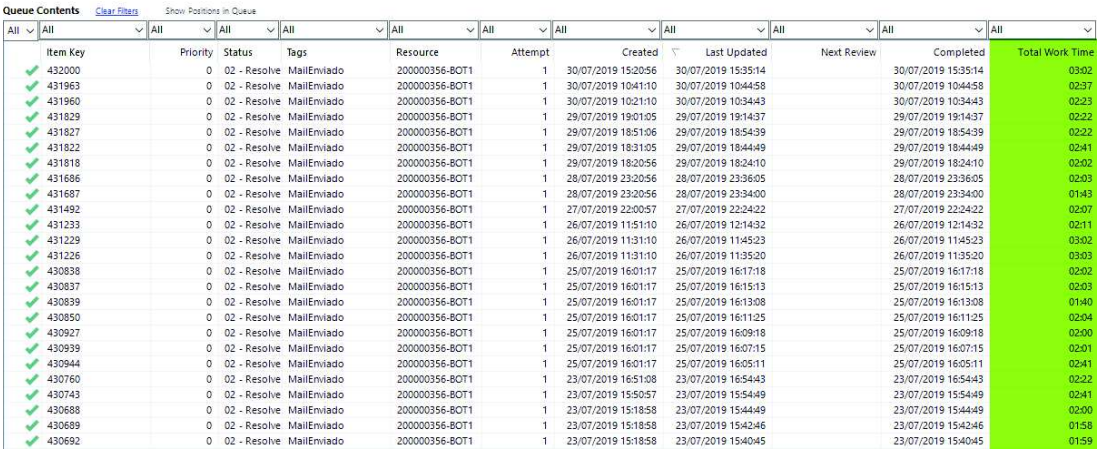


Figura 6.7: Exemplos de execuções do processo *Notify Customers of DoS Attacks with Screenshots*

Quando os analistas SOC estão responsáveis por notificar os clientes, têm que estar disponíveis 24/7. Ou seja, se o analista for notificado de um alerta às 03:00 da manhã, tem a responsabilidade de notificar o cliente independentemente da hora e das condições a que se encontra (por exemplo, estar a dormir).

Para além da disponibilidade, o cuidado a ter com os dados extraídos para compor o *e-mail* de notificação é outra vertente relevante que leva a um aumento da possibilidade de realização de erros durante o processo e ao aumento da duração do processo. De facto, como se apresenta na Figura 6.8, o tempo para cada alerta é superior ao de qualquer alerta representado na Figura 6.5. Contudo, um analista consome entre 300 a 600 segundos desde o início da pesquisa do que pesquisa o alerta no *Arbor* até enviar o *e-mail* de notificação ao cliente. O *robot* demora entre 100 e 200 segundos para realizar as mesmas operações aumentado o desempenho do processo a 200%.

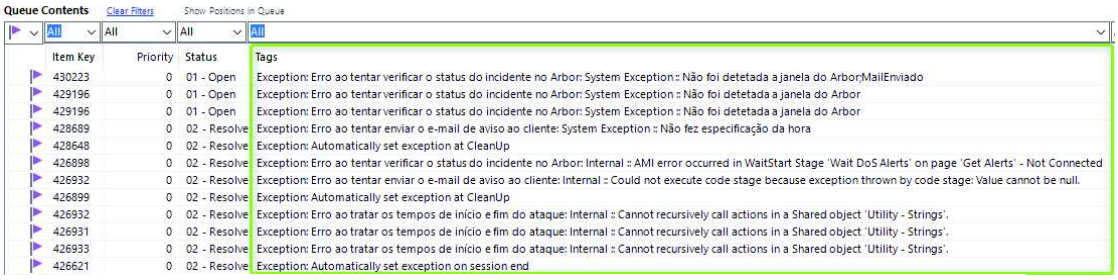


Queue Contents Clear Filters Show Positions in Queue

Item Key	Priority	Status	Tags	Resource	Attempt	Created	Last Updated	Next Review	Completed	Total Work Time
432000	0	02 - Resolve	MailEnviado	200000356-BOT1	1	30/07/2019 15:20:56	30/07/2019 15:35:14		30/07/2019 15:35:14	03:02
431963	0	02 - Resolve	MailEnviado	200000356-BOT1	1	30/07/2019 10:41:10	30/07/2019 10:44:58		30/07/2019 10:44:58	02:37
431960	0	02 - Resolve	MailEnviado	200000356-BOT1	1	30/07/2019 10:21:10	30/07/2019 10:34:43		30/07/2019 10:34:43	02:23
431829	0	02 - Resolve	MailEnviado	200000356-BOT1	1	29/07/2019 19:01:05	29/07/2019 19:14:37		29/07/2019 19:14:37	02:22
431827	0	02 - Resolve	MailEnviado	200000356-BOT1	1	29/07/2019 18:51:06	29/07/2019 18:54:39		29/07/2019 18:54:39	02:22
431822	0	02 - Resolve	MailEnviado	200000356-BOT1	1	29/07/2019 18:31:05	29/07/2019 18:44:49		29/07/2019 18:44:49	02:41
431818	0	02 - Resolve	MailEnviado	200000356-BOT1	1	29/07/2019 18:20:56	29/07/2019 18:24:10		29/07/2019 18:24:10	02:02
431686	0	02 - Resolve	MailEnviado	200000356-BOT1	1	28/07/2019 23:20:56	28/07/2019 23:36:05		28/07/2019 23:36:05	02:03
431687	0	02 - Resolve	MailEnviado	200000356-BOT1	1	28/07/2019 23:20:56	28/07/2019 23:34:00		28/07/2019 23:34:00	01:43
431492	0	02 - Resolve	MailEnviado	200000356-BOT1	1	27/07/2019 22:00:57	27/07/2019 22:24:22		27/07/2019 22:24:22	02:07
431233	0	02 - Resolve	MailEnviado	200000356-BOT1	1	26/07/2019 11:51:10	26/07/2019 12:14:32		26/07/2019 12:14:32	02:11
431229	0	02 - Resolve	MailEnviado	200000356-BOT1	1	26/07/2019 11:31:10	26/07/2019 11:45:23		26/07/2019 11:45:23	03:02
431226	0	02 - Resolve	MailEnviado	200000356-BOT1	1	26/07/2019 11:31:10	26/07/2019 11:35:20		26/07/2019 11:35:20	03:03
430838	0	02 - Resolve	MailEnviado	200000356-BOT1	1	25/07/2019 16:01:17	25/07/2019 16:17:18		25/07/2019 16:17:18	02:02
430837	0	02 - Resolve	MailEnviado	200000356-BOT1	1	25/07/2019 16:01:17	25/07/2019 16:15:13		25/07/2019 16:15:13	02:03
430839	0	02 - Resolve	MailEnviado	200000356-BOT1	1	25/07/2019 16:01:17	25/07/2019 16:13:08		25/07/2019 16:13:08	01:40
430850	0	02 - Resolve	MailEnviado	200000356-BOT1	1	25/07/2019 16:01:17	25/07/2019 16:11:25		25/07/2019 16:11:25	02:04
430927	0	02 - Resolve	MailEnviado	200000356-BOT1	1	25/07/2019 16:01:17	25/07/2019 16:09:18		25/07/2019 16:09:18	02:00
430939	0	02 - Resolve	MailEnviado	200000356-BOT1	1	25/07/2019 16:01:17	25/07/2019 16:07:15		25/07/2019 16:07:15	02:01
430944	0	02 - Resolve	MailEnviado	200000356-BOT1	1	25/07/2019 16:01:17	25/07/2019 16:05:11		25/07/2019 16:05:11	02:41
430760	0	02 - Resolve	MailEnviado	200000356-BOT1	1	23/07/2019 16:51:08	23/07/2019 16:54:43		23/07/2019 16:54:43	02:22
430743	0	02 - Resolve	MailEnviado	200000356-BOT1	1	23/07/2019 15:50:57	23/07/2019 15:54:49		23/07/2019 15:54:49	02:41
430688	0	02 - Resolve	MailEnviado	200000356-BOT1	1	23/07/2019 15:18:58	23/07/2019 15:44:49		23/07/2019 15:44:49	02:00
430689	0	02 - Resolve	MailEnviado	200000356-BOT1	1	23/07/2019 15:18:58	23/07/2019 15:42:46		23/07/2019 15:42:46	01:58
430692	0	02 - Resolve	MailEnviado	200000356-BOT1	1	23/07/2019 15:18:58	23/07/2019 15:40:45		23/07/2019 15:40:45	01:59

Figura 6.8: Representação dos tempos de tratamento de cada alerta na *Queue*

O processo foi posto em produção no dia 28 de Junho de 2019 o que resultou em 2530 execuções e 585 delas resultaram numa exceção. Estas exceções são derivadas a falhas na interação com as aplicações ou por vezes são erros na lógica de implementação. Sempre que é levantada uma exceção, o alerta é marcado como *Exception* na *Queue* e é guardada a descrição da mesma como ilustrado na Figura 6.9.



Queue Contents Clear Filters Show Positions in Queue

Item Key	Priority	Status	Tags
430223	0	01 - Open	Exception: Erro ao tentar verificar o status do incidente no Arbor: System Exception :: Não foi detetada a janela do Arbor;MailEnviado
429196	0	01 - Open	Exception: Erro ao tentar verificar o status do incidente no Arbor: System Exception :: Não foi detetada a janela do Arbor
429196	0	01 - Open	Exception: Erro ao tentar verificar o status do incidente no Arbor: System Exception :: Não foi detetada a janela do Arbor
428689	0	02 - Resolve	Exception: Erro ao tentar enviar o e-mail de aviso ao cliente: System Exception :: Não fez especificação da hora
428648	0	02 - Resolve	Exception: Automatically set exception at CleanUp
426898	0	02 - Resolve	Exception: Erro ao tentar verificar o status do incidente no Arbor: Internal :: AMI error occurred in WaitStart Stage 'Wait DoS Alerts' on page 'Get Alerts' - Not Connected
426932	0	02 - Resolve	Exception: Erro ao tentar enviar o e-mail de aviso ao cliente: Internal :: Could not execute code stage because exception thrown by code stage: Value cannot be null.
426899	0	02 - Resolve	Exception: Automatically set exception at CleanUp
426932	0	02 - Resolve	Exception: Erro ao tratar os tempos de início e fim do ataque: Internal :: Cannot recursively call actions in a Shared object 'Utility - Strings'.
426931	0	02 - Resolve	Exception: Erro ao tratar os tempos de início e fim do ataque: Internal :: Cannot recursively call actions in a Shared object 'Utility - Strings'.
426933	0	02 - Resolve	Exception: Erro ao tratar os tempos de início e fim do ataque: Internal :: Cannot recursively call actions in a Shared object 'Utility - Strings'.
426621	0	02 - Resolve	Exception: Automatically set exception on session end

Figura 6.9: Exemplos de exceções na execução do caso de uso

Visto que o processo foi posto em produção há relativamente pouco tempo, não foi possível calcular o valor de elementos tratados. Porém, pode-se afirmar que o volume de elementos tratados é três vezes mais rápido face aos analistas. E com a rapidez adquirida através da automação, o objetivo de notificar o cliente o mais rápido possível foi atingido com sucesso tanto que, o máximo tempo possível que um alerta pode levar a ser notificado é 10 minutos.

Deste modo, a Tabela 8 apresenta o resumo da avaliação da execução do processo entre os analistas SOC e o robot:

Critérios	Analista SOC	Robot
Tempo de execução	300 - 600 Segundos	100 - 200 Segundos
Percentagem de taxa de erro	*	≈ 23.1%
Disponibilidade	?	100%

? – Possibilidade do analista estar doente ou de férias ou ser feriado

* – Não é possível calcular

Tabela 8: Valores das métricas para o analista e o robot do caso de uso *Notify Customers of DoS Attacks with Screenshots*

Com a Tabela 8 verificamos que a taxa de erro é considerável, contudo são erros que podem ser corrigidos. A disponibilidade é total, o que quer dizer que para situações de exceção, todos os alertas que ficaram por tratar serão cobertos na próxima execução. Ou seja, os analistas não precisam mais de verificar os alertas para notificar os clientes.

Capítulo 7
Conclusão

Ao longo da realização do projeto, o objetivo foi implementar processos da área do SOC da MEO com ferramentas de automação, colocá-los em produção e disponibilizar o tempo dos analistas para concretizarem outros processos que requerem um nível de cognição superior e poder de decisão. Para além disso, com a automação, recebe-se outros ganhos como o aumento no desempenho e a diminuição da taxa de erro.

Apesar das adversidades relacionadas com a ferramenta de SOAR (*ATAR Labs*), foi possível implementar a maior parte dos casos propostos com a ferramenta de RPA (*Blue Prism*). Tal como a avaliação das ferramentas de RPA no mercado, o *Automation Anywhere Enterprise* e o *Blue Prism* satisfazem as expectativas na implementação de processos. A linha de aprendizagem cresce rapidamente ao longo do tempo visto que são tecnologias intuitivas. Para além disso, o modo de agendamento dos processos é simples e dá a possibilidade de saber facilmente organizar a agenda do robot de modo a que esteja sempre disponível evitando a sobreposição de processos.

Tendo em conta o tempo de realização do projeto, pode-se concluir que a implementação de todos os robots foi feita rapidamente. A etapa que durou mais foi a fase de testes que foi necessário pensar em todas as situações indesejadas que levavam ao levantamento de exceções e criar um caminho alternativo para que os processos executassem sem erros. Para além disso, o modo de implementação é simples visto que são usados diagramas de fluxo e simplifica a leitura do "código" para um leitor externo. Outra vantagem encontrada foi que as ferramentas de RPA posicionam-se como ferramentas próprias para processos *business*, mas com o projeto realizado conclui-se que também são flexíveis o suficiente para possibilitarem a implementação de processos de negócios pertencentes ao ambiente de ciber segurança. Um exemplo de um processo de negócio repetitivo de ciber segurança é o caso de uso responsável por notificar os clientes dos ataques de *Denial of Service*.

Relativamente aos resultados apresentados, podem ser feitas algumas conclusões como a disponibilidade do robot ser superior face à dos analistas. Ao contrário dos robots, os analistas cansam-se após a realização de inúmeras tarefas repetitivas. Outras variáveis não contabilizadas (para além do cansaço) são: o analista faltar por doença ou estar de férias. Este tipo de situações podem retardar o trabalho e por outro lado, impede-os de concretizar tarefas de maior relevância no SOC. Deste modo, automatizando os processos rotineiros no SOC é oferecido mais tempo aos analistas para concretizarem as tarefas mais complexas e por consequência, leva-os a ter mais gosto na execução do seu exercício e não desgasta cognitivamente.

Por outro lado, os resultados das métricas apresentadas no Capítulo 6 não são totalmente fiáveis porque o tempo de observação dos robots em produção é curto, e por esse motivo a métrica, como a percentagem de taxa de erro, apresentou valores elevados face ao que era expectável. O maior benefício encontrado é o aumento do desempenho nos processos e por consequência a produtividade.

Tal como mencionado, o foco principal do projeto era automatizar algumas tarefas repetitivas do SOC para aumentar a disponibilidade dos analistas para realizarem outras tarefas menos rotineiras e mais complexas. Com base no trabalho apresentado, pode-se concluir que o objetivo foi cumprido com sucesso para alguns casos de uso.

Para trabalho futuro, prevê-se fazer melhorias na lógica dos processos apresentados para diminuir o número de erros possíveis na execução dos mesmos. Outro objetivo é criar o ambiente SOC correto para usar o *ATAR Labs*, *i.e.*, fazer a migração dos clientes para o novo SIEM *QRadar* e integrá-lo na

ferramenta de SOAR. Após a adição da ferramenta *ATAR Labs* implementa-se o caso de uso *Verify Brute-Force Attacks in LoginPT* e outros processos do SOC.

Outro objetivo para trabalho futuro é conhecer melhor os restantes componentes que não foram usados das ferramentas *Blue Prism* e *ATAR Labs* e complementar o conhecimento já adquirido destas tecnologias (através do projeto) acompanhando as evoluções das mesmas. De momento, as ferramentas de RPA estão a evoluir para um nível mais sofisticado onde para além da tomada de decisão a partir de regras querem adicionar Inteligência Artificial ao robot para ter algum nível cognitivo e conseguir realizar tarefas rotineiras de maior complexidade, como por exemplo automatizar a mitigação de ataques ocorrentes no SOC da MEO.

Referências

Campbell, Donald (1988). *Task Complexity: A Review and Analysis. Academy of Management Review*, Vol. 13, 40-52.

Larman, Craig (Outubro 2004). *Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development*, Third Edition. Chapter 10.

Kappagantula, Sahiti (acedido em Agosto 2019). *RPA Tools List and Comparison – Leaders in RPA Software*. https://www.edureka.co/blog/rpa-tools-list-and-comparison/.

Le Clair, Craig (Junho 2018). *The 15 Providers That Matter Most And How They Stack Up*. https://samfundsdesign.dk/siteassets/media/downloads/pdf/the_forrester_wave_rpa_2018_uipath_rpa_leader.pdf

Automation Anywhere Bot Store (acedido em Maio 2019). https://botstore.automationanywhere.com/bot/automated-invoice-processing-using-iq-bot/.

Automation Anywhere Bot Store (acedido em Maio 2019). https://developer.ibm.com/answers/storage/attachments/26271-screenshot-24.png.

Automation Anywhere Bot Store (acedido em Maio 2019). https://botstore.automationanywhere.com/bot/retrieve-user-credentials-from-secure-vault-2/.

Digital Workforce (acedido em Julho 2019). *Blue Prism CTO, Dave Moss answers beginner's questions about Robotic Process Automation*. https://digitalworkforce.com/rpa-news/blue-prism-cto-dave-moss-answers-beginners-questions-about-robotic-process-automation/.

Jacobs, Pierre, Alapan Arnab and Barry Irwin. *Classification of Security Operation Centers*. 2013 Information Security for South Africa. IEEE, 2013.

Madani, Afsaneh, Saed Rezayi and Hossein Gharace. *Log Management comprehensive architecture in Security Operation Center(SOC)*. International Conference on Computational Aspects of Social Networks (CASoN), 2011.

Jain, Jatin (acedido em Julho 2019). *What Is a SIEM?* InfoSEc Institute https://resources.infosecinstitute.com/what-is-a-siem/.

Micro Focus Community (acedido em Julho 2019). *What is False Positive, false negative, true positive and true negative?* https://community.microfocus.com/t5/ArcSight-User-Discussions/what-is-false-positive-false-negetive-true-positive-and-true/td-p/1582039.

InfoCyte (acedido em Julho de 2019). *Cybersecurity 101: What You Need To Know About False Positives.* https://community.microfocus.com/t5/ArcSight-User-Discussions/what-is-false-positive-false-negetive-true-positive-and-true/td-p/1582039

Lau, Felix, Stuart Rubin, Michael Smith and Ljiljana Trajkovic. *Distributed Denial of Service Attacks.* IEEE, 2000.

Kácha, Pavel (Setembro 2009). *Adapting Ticket Request System to the Needs of CSIRT Teams.* CESNET-CERTS Computer Security Incident Response Team. https://dl.acm.org/citation.cfm?id=1718109.1718112.

Automation Anywhere (acedido em Maio 2019). https://www.automationanywhere.com/company/about-us.

Dierenfeld, Helena and Agathe Merceron (Setembro 2012*). Learning Analytics with Excel Pivot Tables*. Conference Proceedings, Research Conference.

Stallings, William. *Cryptography and Network Security: Principles and Practice*. Fifth Edition, Chapter 5 (147-191).

International Organization for Standardization. Date and Time Forma – ISO 8601 (acedido em Julho 2019). https://www.iso.org/iso-8601-date-and-time-format.html.

Youtube (acedido em Julho 2019). *White Collar Robots: The Virtual Workforce | David Moss | TEDxUCL*. https://www.youtube.com/watch?v=1SximAg9t4w&feature=youtu.be

Aguirre, Santiago and Alejandro Rodriguez (2017). *Automation of a Business Process Using Robotic Process Automation (RPA): A Case Study*. Springer International Publishing, 2017.

Oshin, Mayo (acedido em Agosto 2019). *How the Productivity Formula Can Motivate Employees to Work Effeciently*. Lifehack. https://www.lifehack.org/808717/productivity-formula

Fung, Han Ping. *Criteria, Use Cases and Effects of Information Technology Process Automation (ITPA)*. Advances in Robotics & Automation 3 (2014).

Boulton, Clint (acedido em Agosto 2019). *What is RPA? A revolution in business process automation*. CIO. https://www.cio.com/article/3236451/what-is-rpa-robotic-process-automation-explained.html.

Correia, Luis Oliveira (acedido em Agosto 2019). *Robotic Process Automation*. O Jornal Económico. https://jornaleconomico.sapo.pt/noticias/robotics-process-automation-110272.

Su, Jeb (acedido em Agosto 2019). *Why SoftBank Invested $300 Million In Robotic Process Automation (RPA)*. Forbes. https://www.forbes.com/sites/jeanbaptiste/2019/07/03/why-softbank-invested-300-million-in-robotic-process-automation-rpa/#6b8c141b7e6f

Davenport, Tom (acedido em Agosto 2019). *A Marriage of Robotic Process Automation and Machine Learning. Forbes*. https://www.forbes.com/sites/tomdavenport/2019/06/06/a-marriage-of-robotic-process-automation-and-machine-learning/#450357ef7947.

Davenport, Tom (acedido em Agosto 2019). *Can Process Robots Deliver Digital Transformation?* Forbes. https://www.forbes.com/sites/tomdavenport/2017/09/20/can-process-robots-deliver-digitaltransformation/#2414240ece75.

Chen, James (acedido em Agosto 2019). *Return of Investment (ROI)*. Investopedia. https://www.investopedia.com/terms/r/returnoninvestment.asp.

Miller, Ron (acedido em Agosto 2019). *Gartner finds RPA is fastest growing market in enterprise software. Techrunch.* https://techcrunch.com/2019/06/24/gartner-finds-rpa-is-fastest-growing-market-in-enterprise-software/.

Gartner (acedido em Agosto 2019). *Gartner Says Worldwide Robotic Process Automation Software Market Grew 63% in 2018*. https://www.gartner.com/en/newsroom/press-releases/2019-06-24-gartner-says-worldwide-robotic-process-automation-sof.

Le Clair, Craig (acedido em Agosto 2019). *How Automation Is Impacting Enterprises in 2019*. Forrester Wave. https://go.forrester.com/blogs/predictions-2019-automation-technology/.

Balaban, David (acedido em Agosto 2019). *An Overview of UBA, SIEM and SOAR solutions: What Are the Differences?* SynexIntroduces. https://n0where.net/an-overview-of-uba-siem-and-soar-solutions-what-are-the-differences.

Gartner IT Glossary (acedido em Agosto 2019). *Security Orchestration Automation and Response (SOAR).* https://www.gartner.com/it-glossary/security-orchestration-automation-response-soar.

Rouse, Margaret (acedido em Agosto 2019). *Definition SOAR (Security Orchestration, Automation and Response)*. Search Security. https://searchsecurity.techtarget.com/definition/SOAR.

InfoSec (acedido em Agosto 2019). *Security Orchestration, Automation and Response (SOAR)*. https://resources.infosecinstitute.com/security-orchestration-automation-and-response-soar/#gref.

Avast (acedido em Agosto 2019). *Phishing: O que é phising*. https://www.avast.com/pt-br/c-phishing.

Moran, John (acedido em Agosto 2019). *The State of Security Orchestration Automation and Response (SOAR) in 2019*. https://www.dflabs.com/blog/the-state-of-security-orchestration-automation-and-response-soar-in-2019/.

Michelson, Brenda (Fevereiro 2006). *Event-Driven Architecture Overview*. Patricia Seybold Group.

Montesino, Raydel and Stefan Fenz. *Information security automation: how far can we go?* 2011 Sixth International Conference on Availability, Reliability and Security.

Saurabh, Kumar (acedido em Agosto 2019). *How cognitive and robotic automation play in SecOps.* https://www.csoonline.com/article/3207534/how-cognitive-and-robotic-automation-play-in-secops.html.

Akinjayeju, Alexander (acedido em Agosto 2019). *Robotic Process Automation & AI in Cyber Security*. https://jayeju.com/automation-artificial-intelligence-in-security-operations/.

CiGen (acedido em Agosto 2019). *Why Robotic Process Automation (RPA) is Your Ally in Reducing Cyber Security Risks*. https://www.cigen.com.au/cigenblog/robotic-process-automation-rpa-your-ally-reducing-cyber-security-risks.

Phillippe, James, Rohit Rao, Alex Campbell and Satoshi Fujii-san (2018). *How do you protect the robots from cyber attack?* EY

AI Multiple (acedido em Agosto 2019). *57 RPA Use Cases/Applications/Examples: In-Depth Guide [2019]*. https://blog.aimultiple.com/robotic-process-automation-use-cases/

Wazlawick, Raul (acedido em Agosto 2019). *Learn more about System Sequence Diagram.* https://www.sciencedirect.com/topics/computer-science/system-sequence-diagram. *ScienceDirect*.

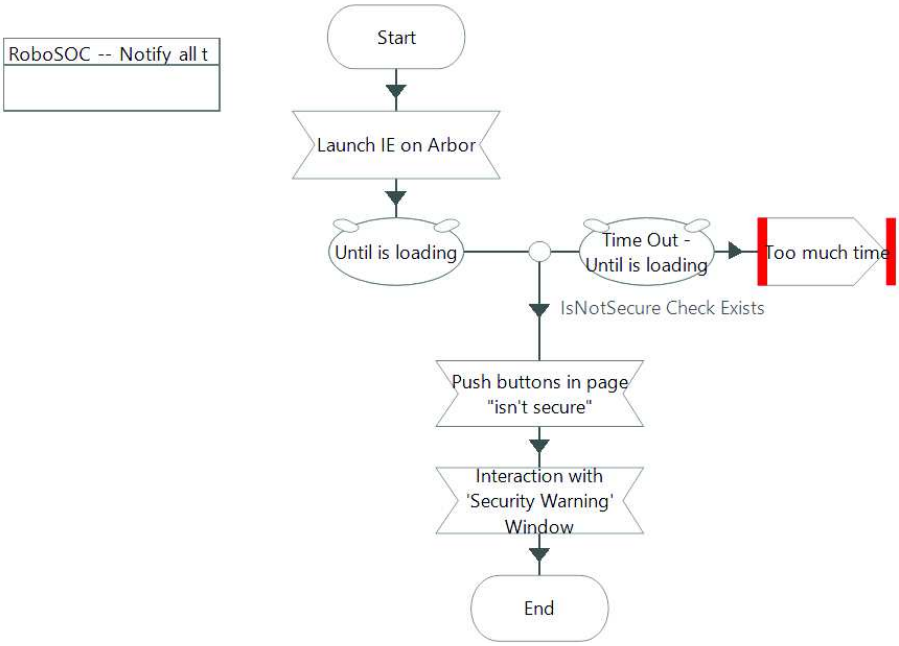
Lacity, Mary, Leslie Willcocks and Andrew Craig (Abril 2015). *Robotic Process Automation at Telefínica O2*.

*Brocke, Jan vom, Wolfgang Maa*β, Peter Buxmann, Alexander Maedche, Jan Macro Leimeister and Günter Pecht (Maio 2018). *Future Work and Enterprise Systems*. Springer, 2018. https://link.springer.com/article/10.1007%2Fs12599-018-0544-2.

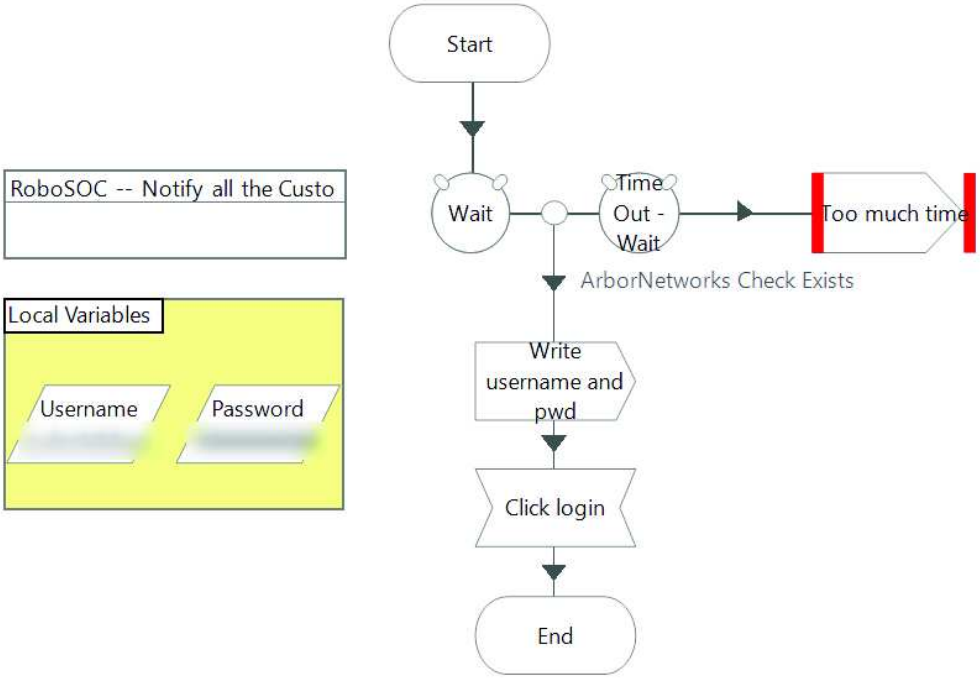
Expert System (acedido em Setembro 2019). *What is Machine Learning? A definition*. https://www.expertsystem.com/machine-learning-definition/.

Apêndice A *Nofity Customers of DoS Attacks with Reports* (Primeira Iteração)

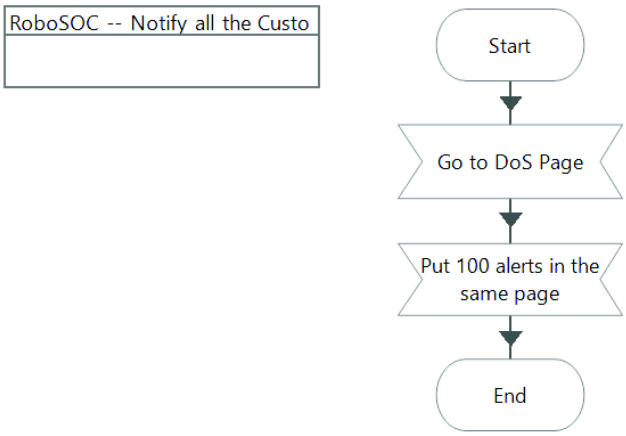
A.1 *Launch Google Chrome and Arbor* (*Object*)



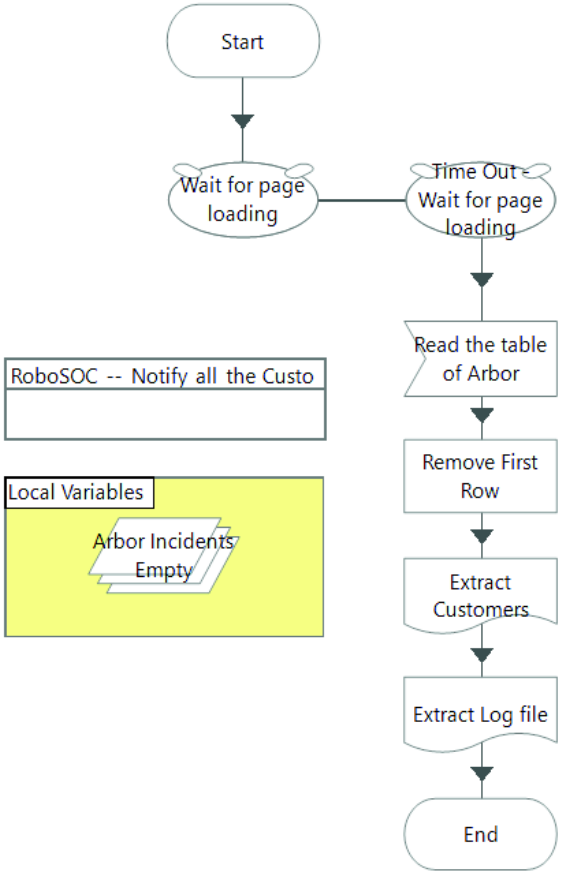
A.2 *Login Arbor* (*Object*)



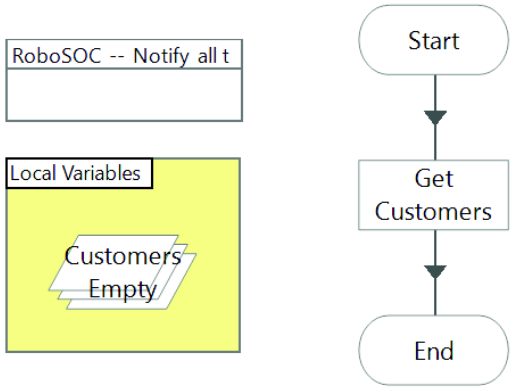
A.3 *Go to DoS Page and Filter Incidents* (*Object*)



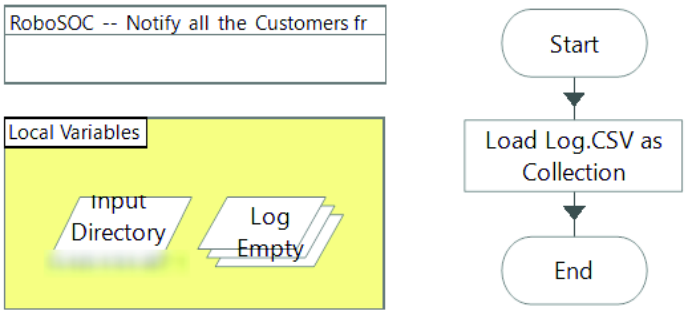
A.4 *Get all Incidents* (*Object*)



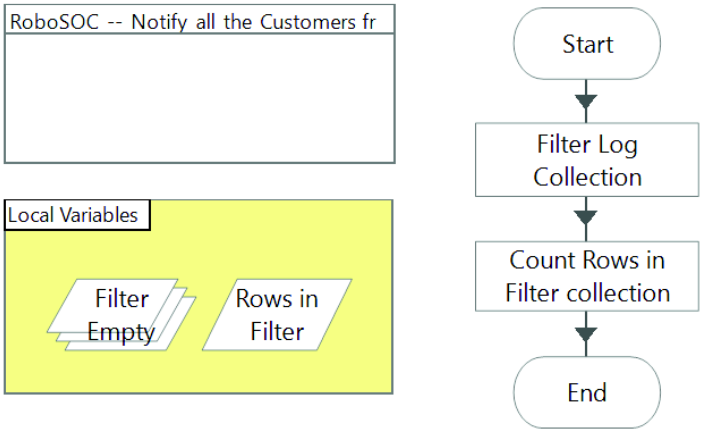
A.5 *Get Customers List* (*Object*)



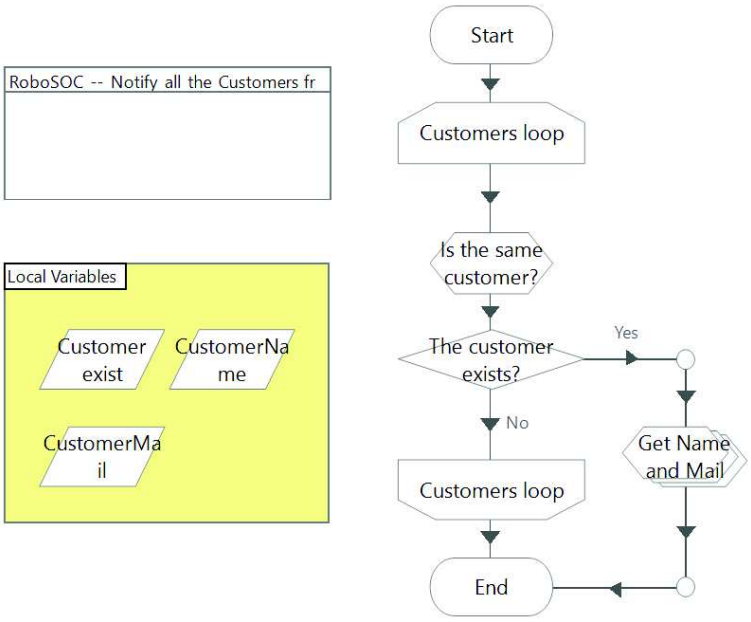
A.6 *Get Log* (*Object*)



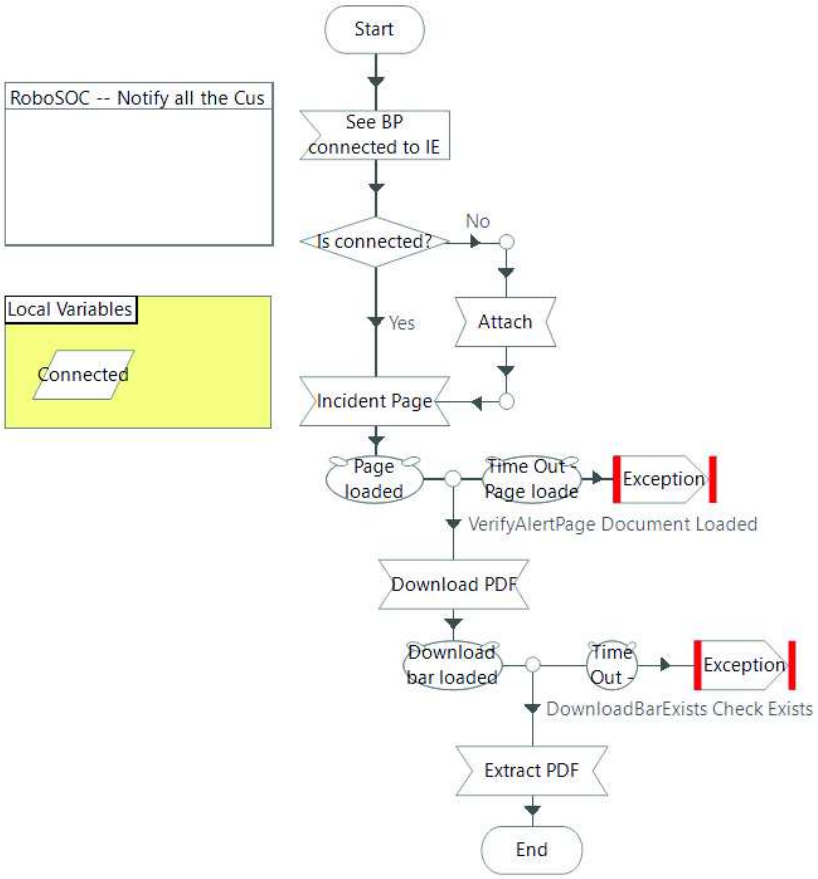
A.7 *Filter Log* (*Object*)



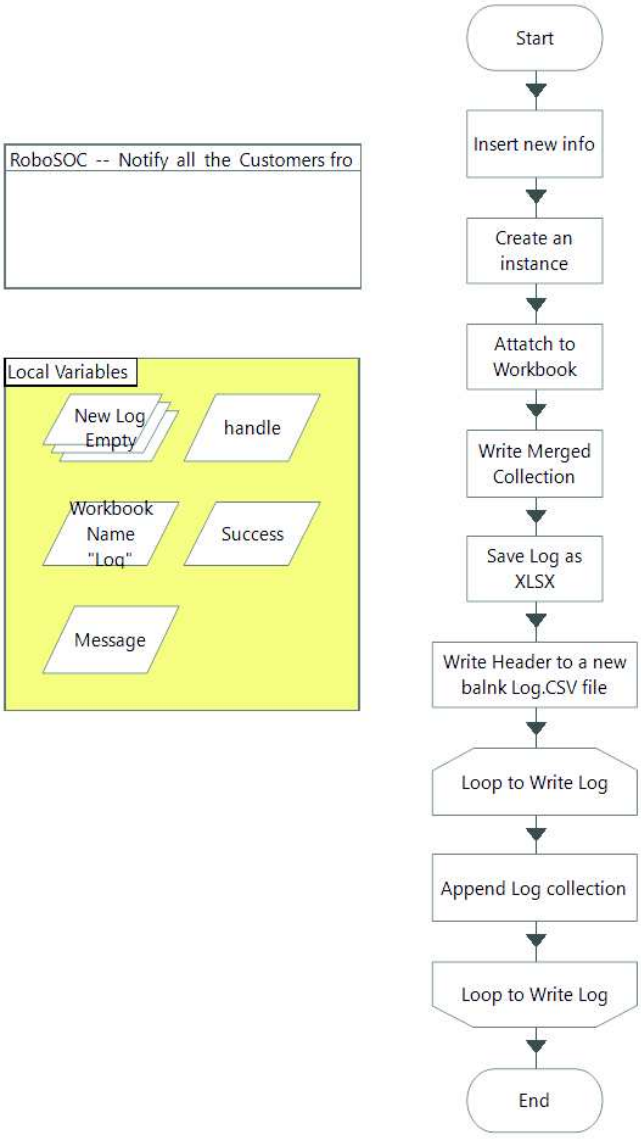
A.8 *Find Customer* (*Object*)



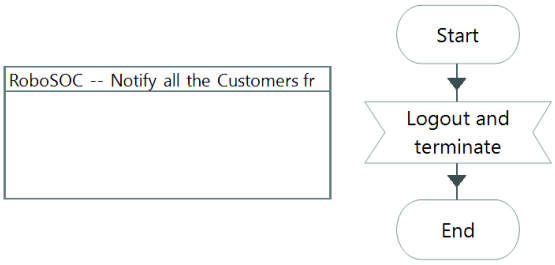
A.9 *Extract Incident Report* (*Object*)



A.10 *Create the new log file* (*Object*)

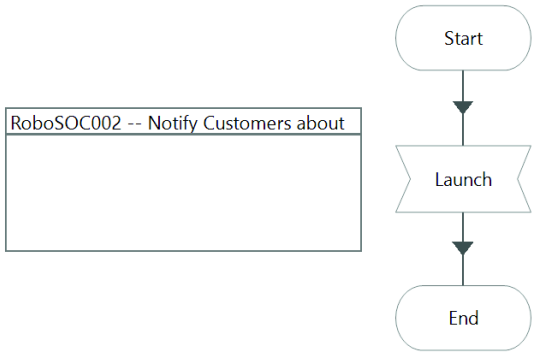


A.11 *Logout Arbor and terminate Google Chrome* (*Object*)

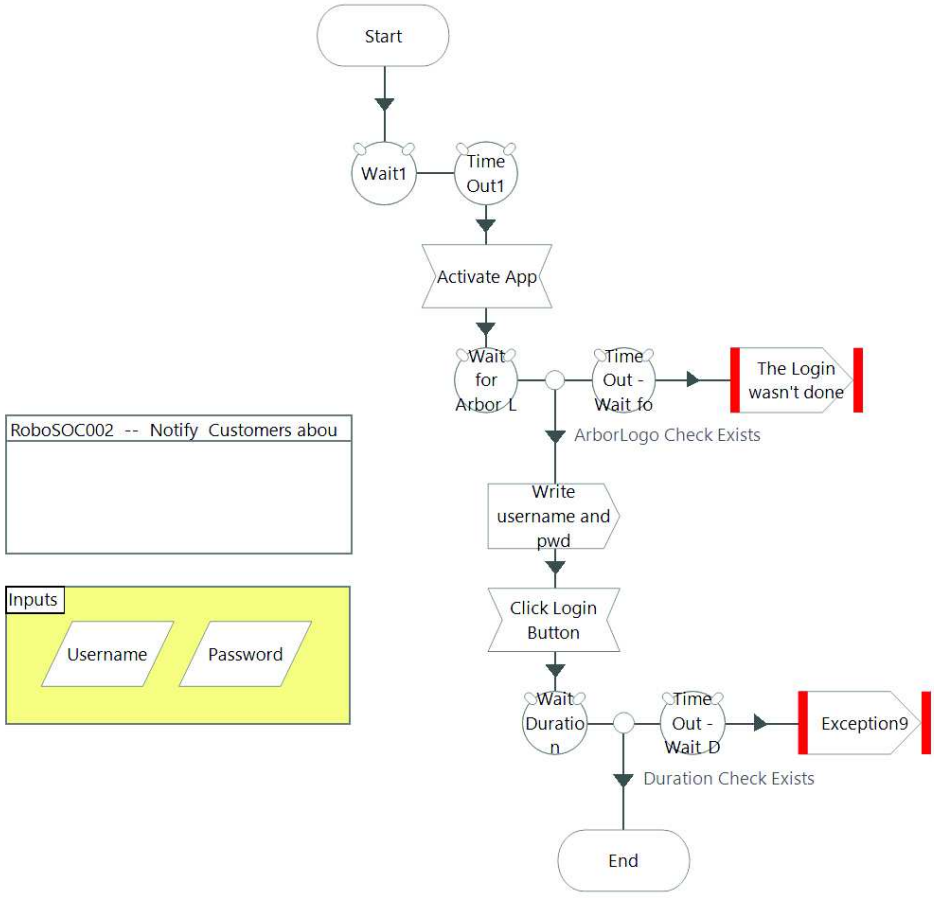


Apêndice B *Nofity Customers of DoS Attacks with Reports* (Segunda Iteração)

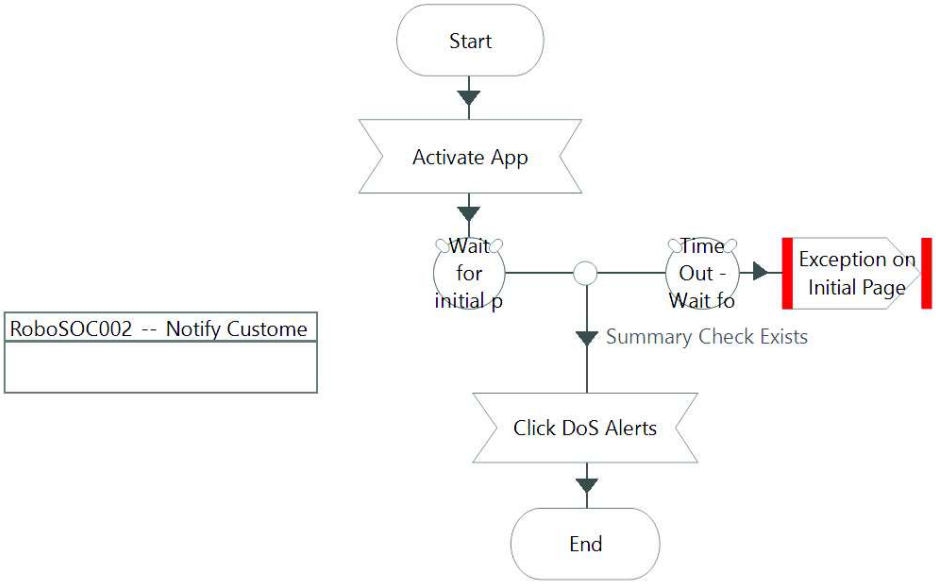
B.1 *Launch Arbor* (*Object*)



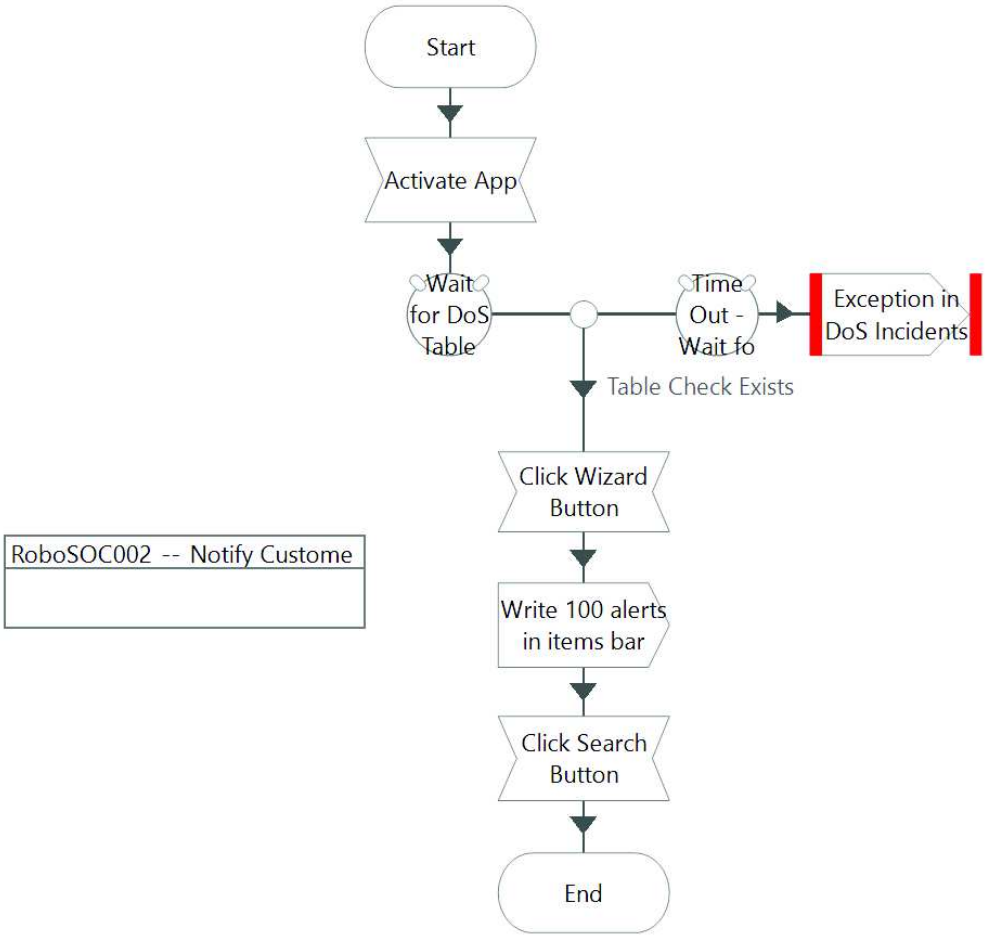
B.2 *Login Arbor* (*Object*)



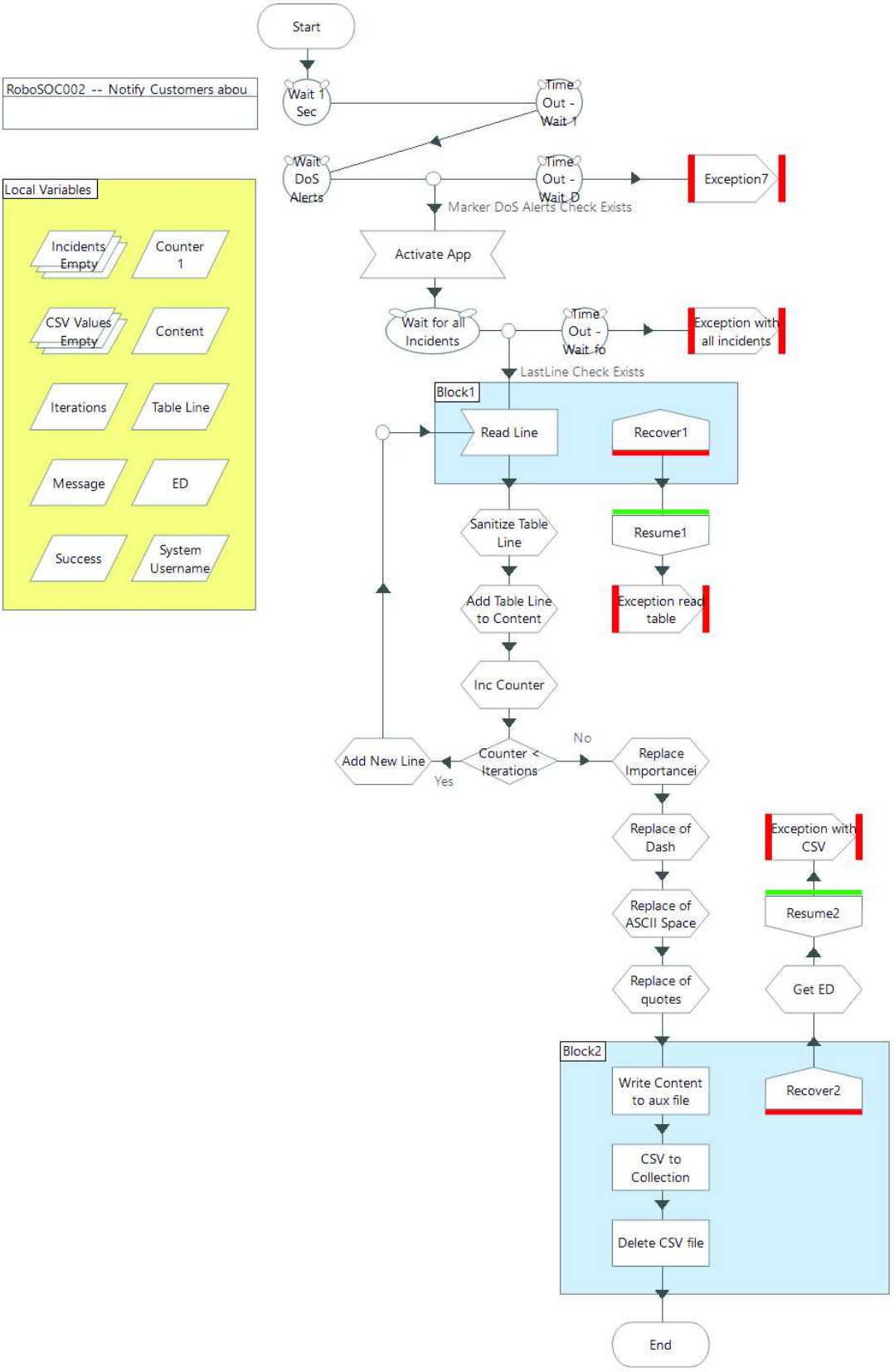
B.3 *Go to DoS Page* (*Object*)



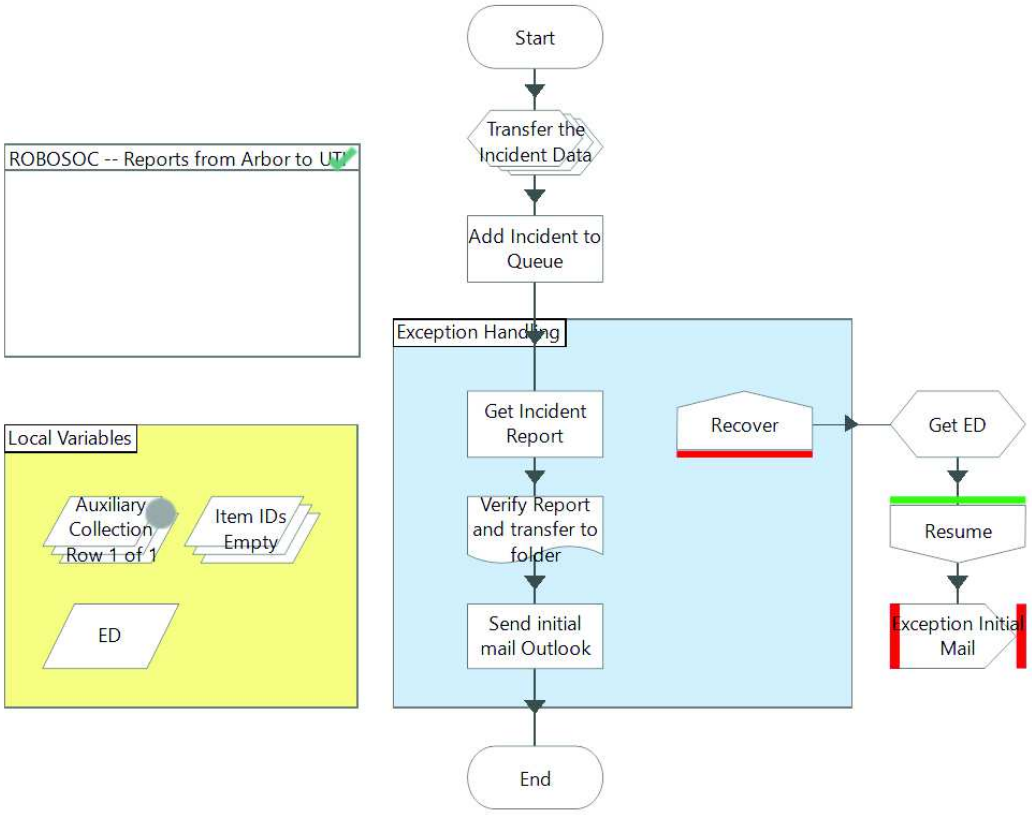
B.4 *Filter Incidents* (*Object*)



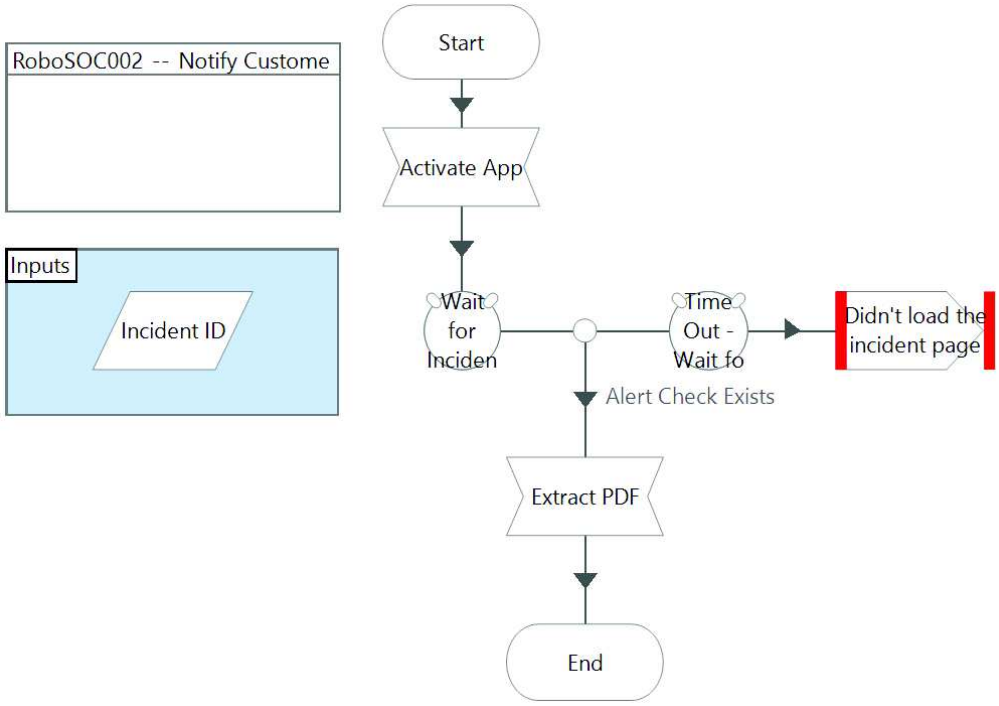
B.5 *Get all Incidents (Object)*



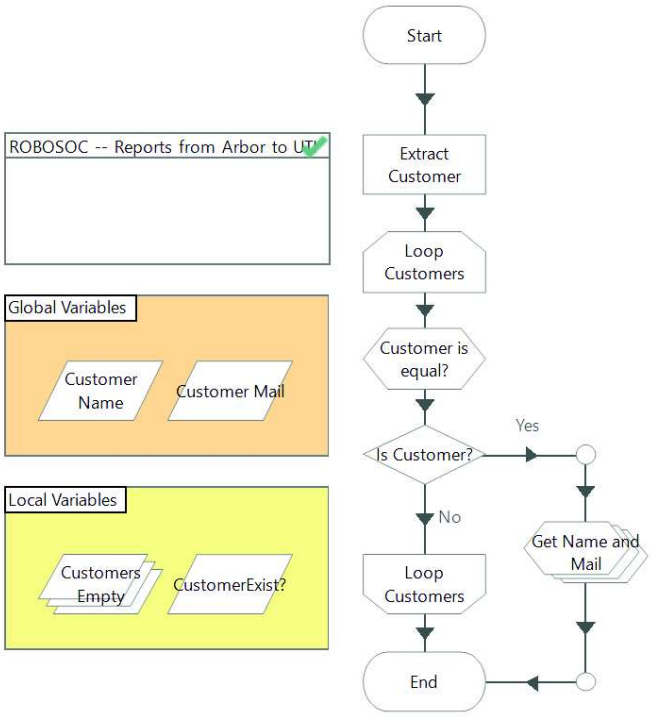
B.6 *Send Initial Email* (*Process*)



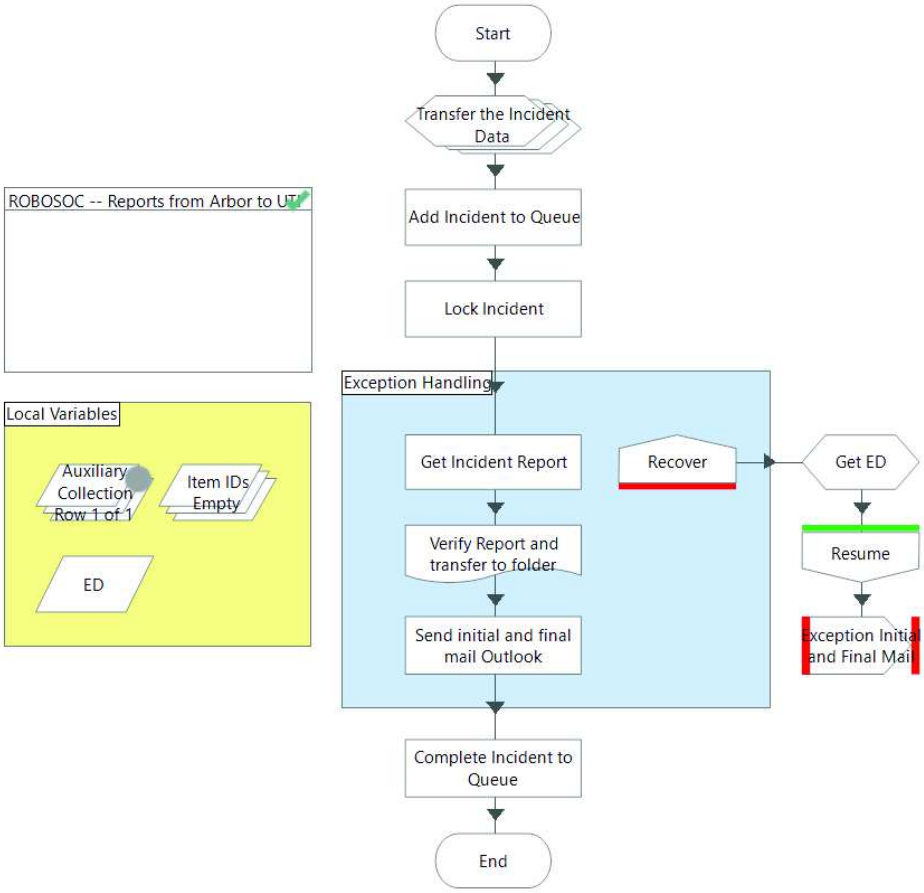
B.7 *Get Incident Report* (*Object*)



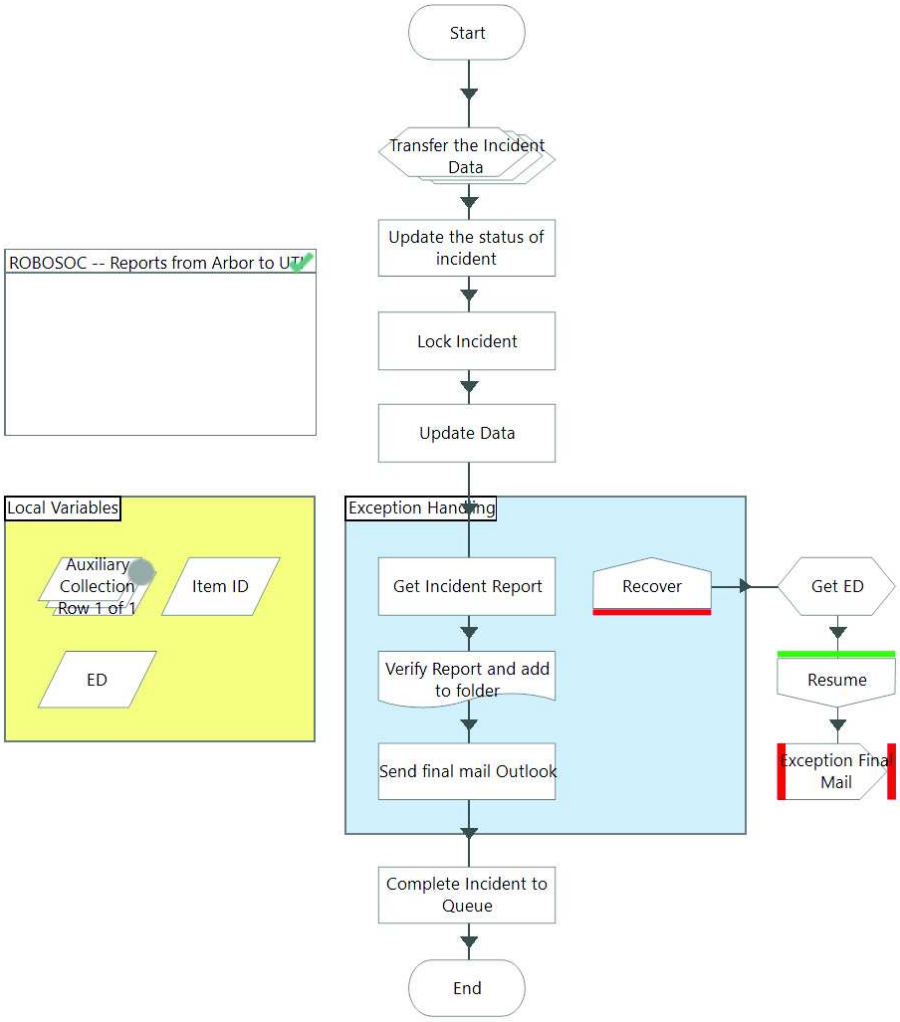
B.8 *Verify Customer* (*Process*)



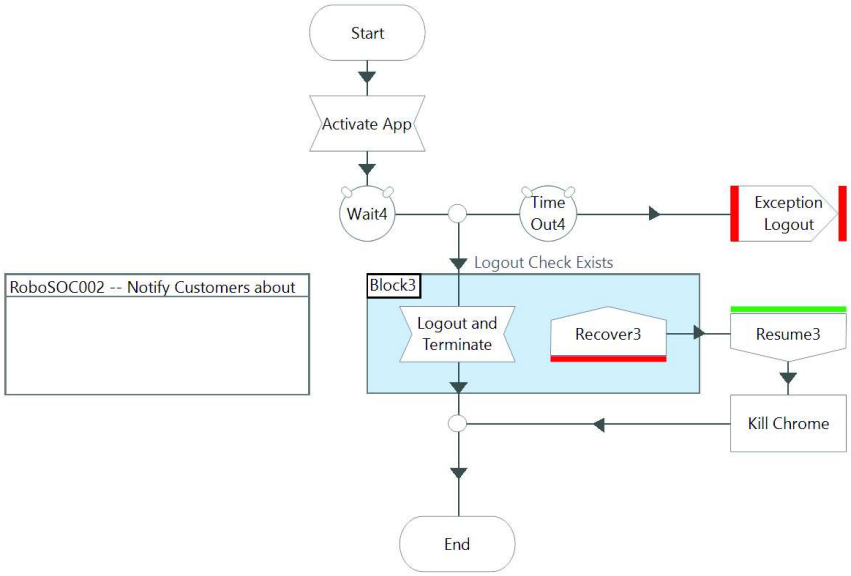
B.9 *Send Initial and Final Email* (*Process*)



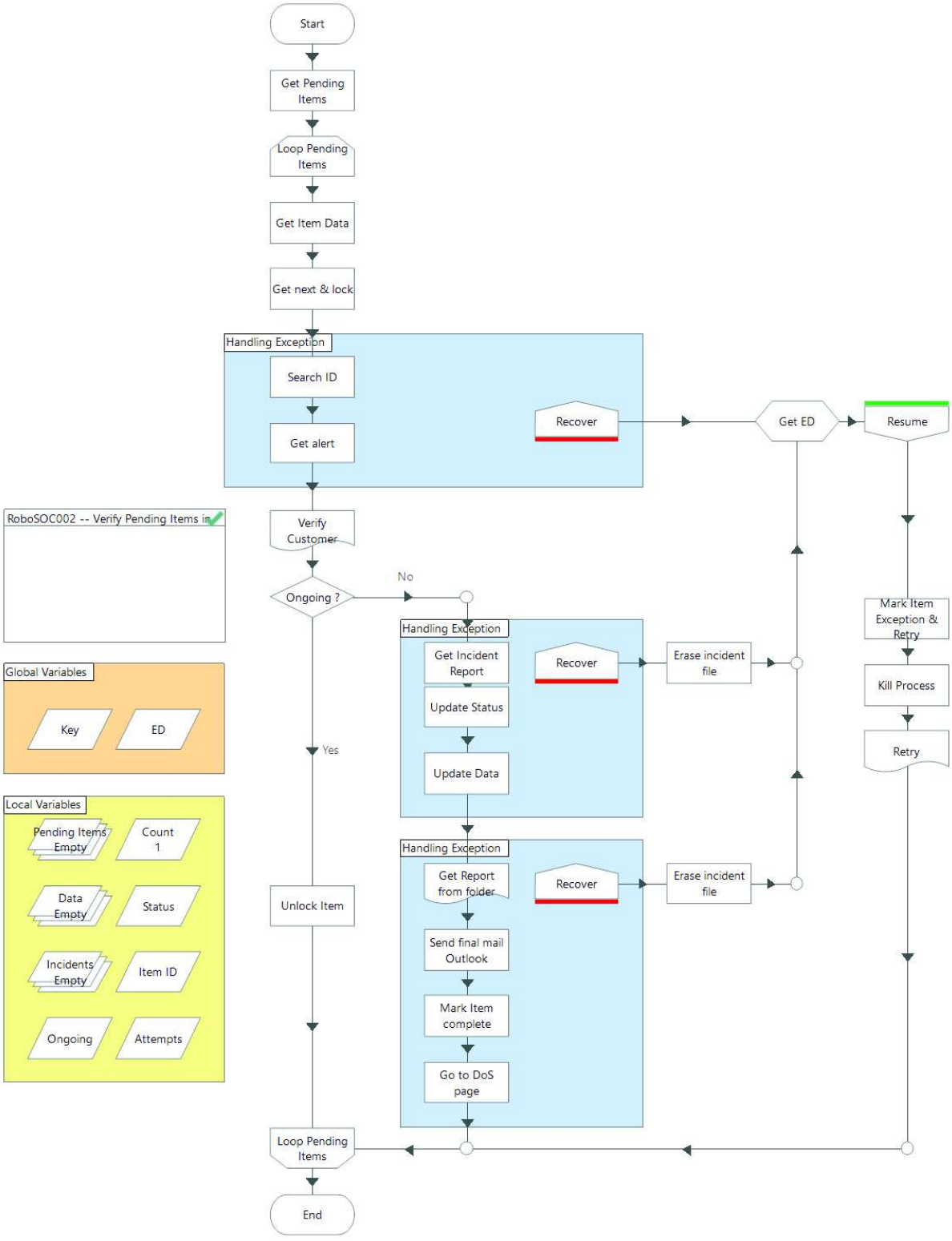
B.10 *Send Final Email* (*Process*)



B.11 *Logout Arbor and Terminate Chrome* (*Object*)

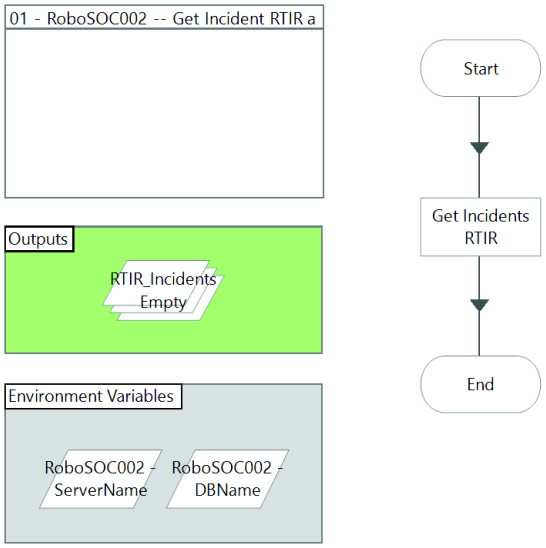


B.12 *Get Pending Items* (*Process*)

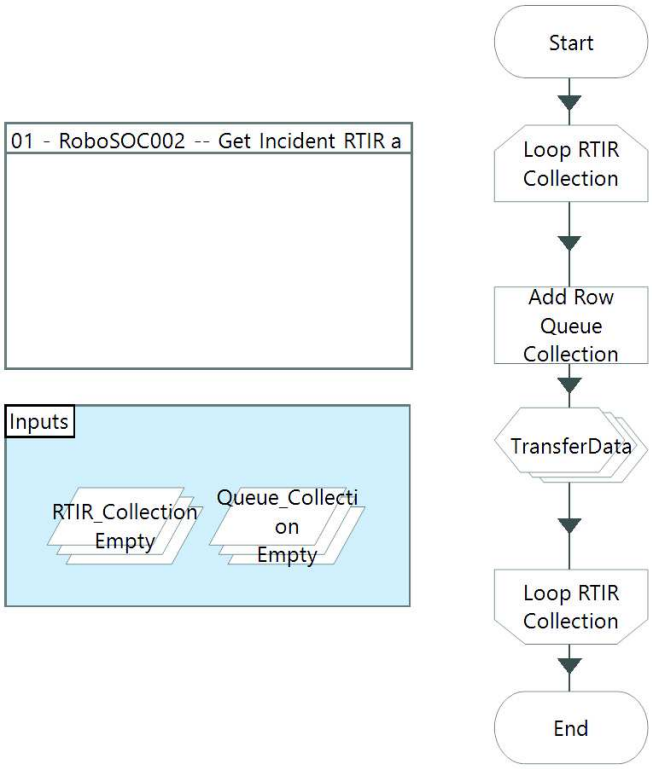


Apêndice C *Notify Customers of DoS Attacks with Screenshots*

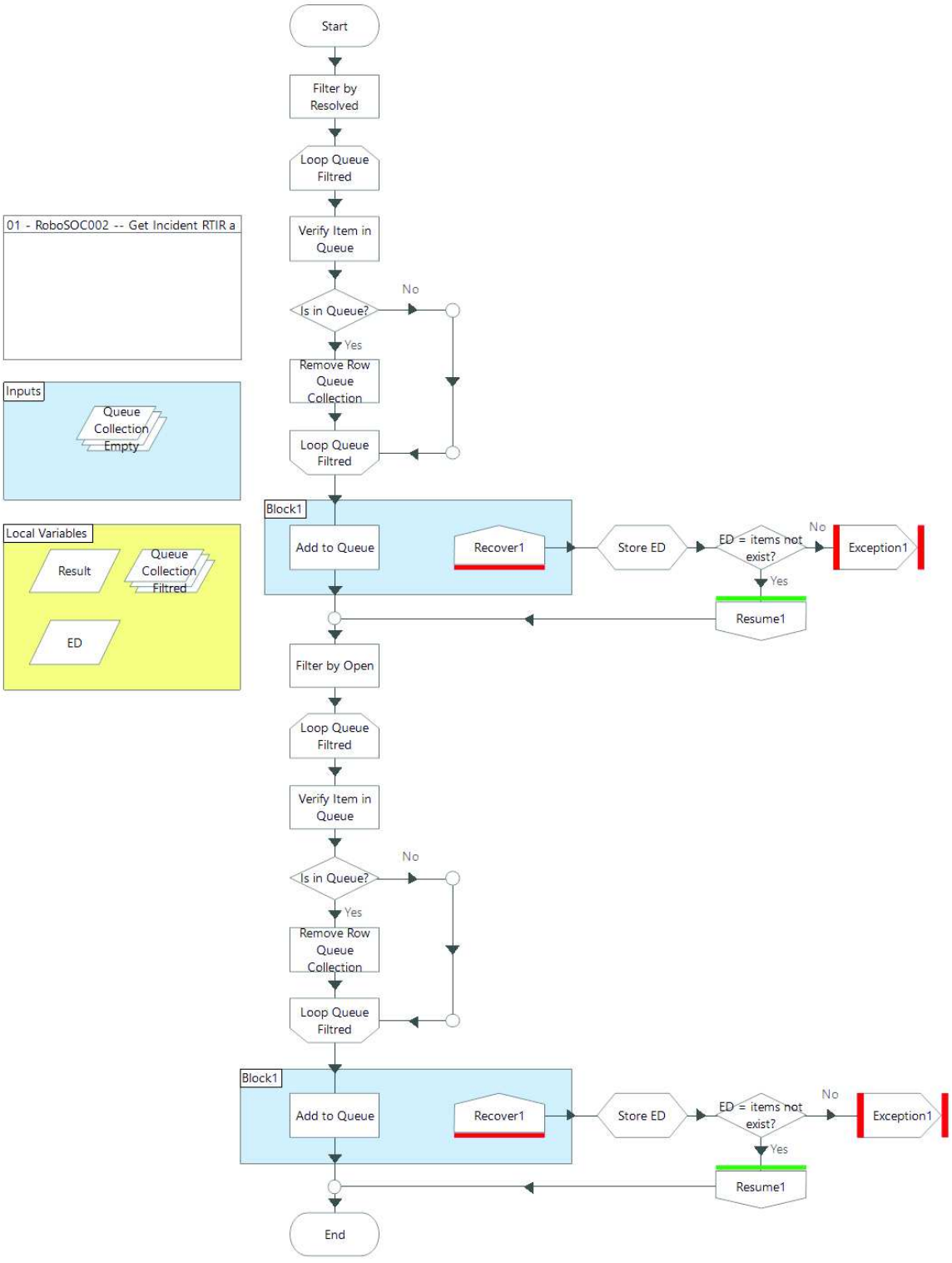
C.1 *Get RTIR Incidents* (*Process*)



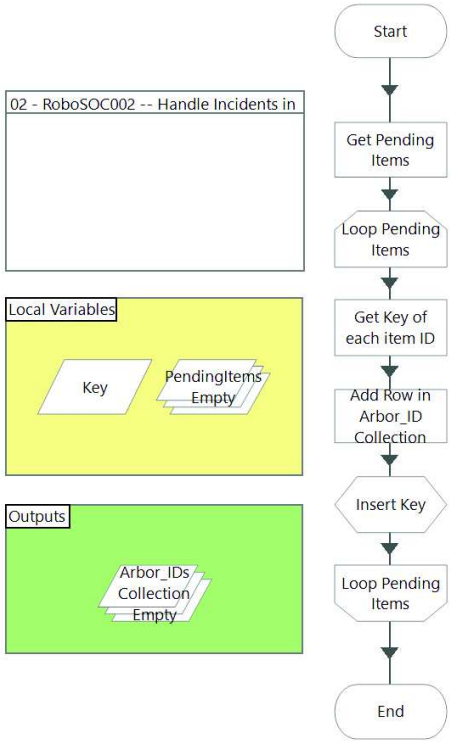
C.2 *RTIR to Queue Collection* (*Process*)



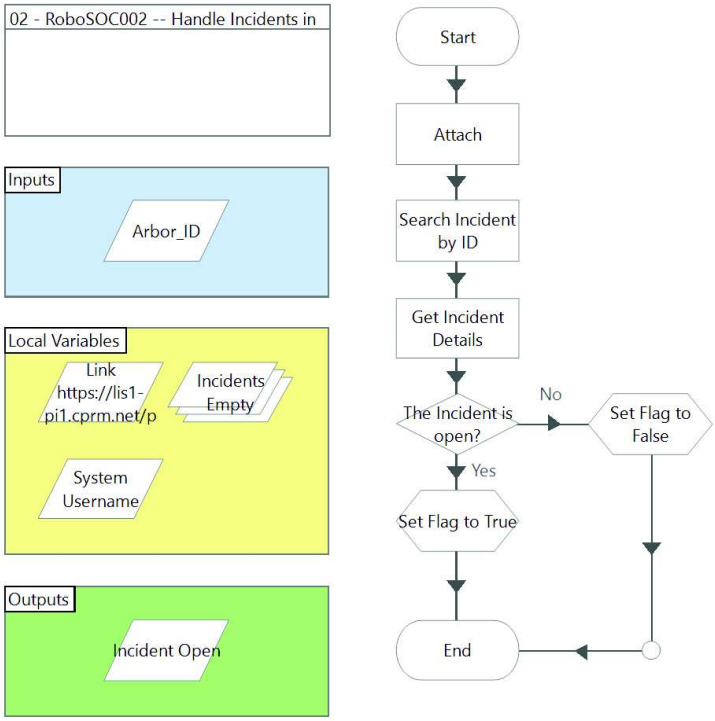
C.3 *Add to Queue* (*Process*)



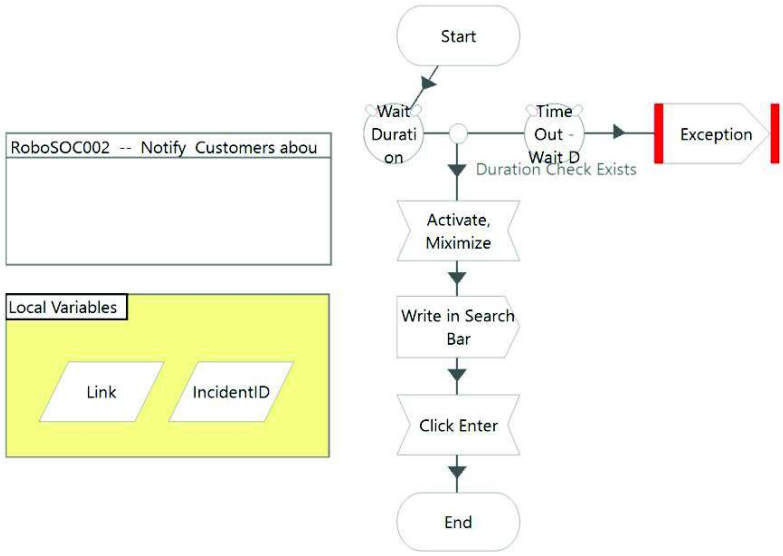
C.4 *Get Pending Items* (*Process*)



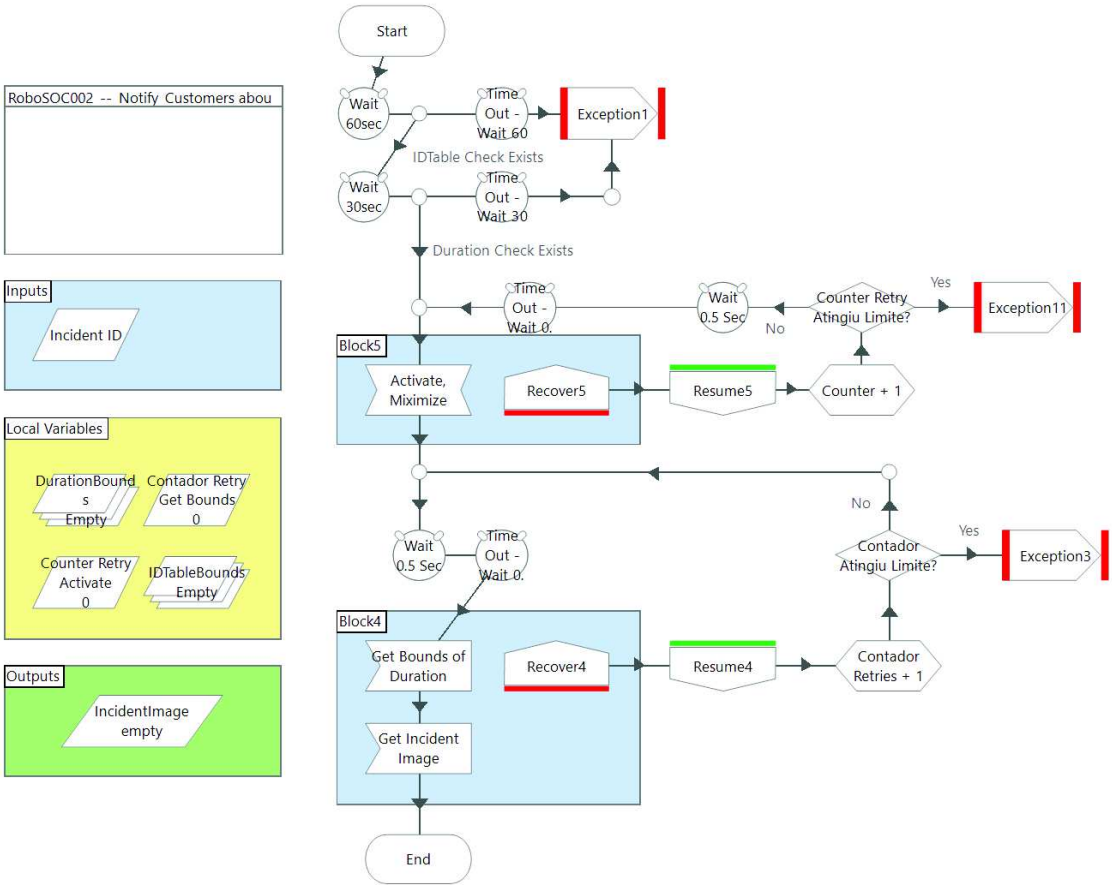
C.5 *Go To Incident Page* (*Process*)



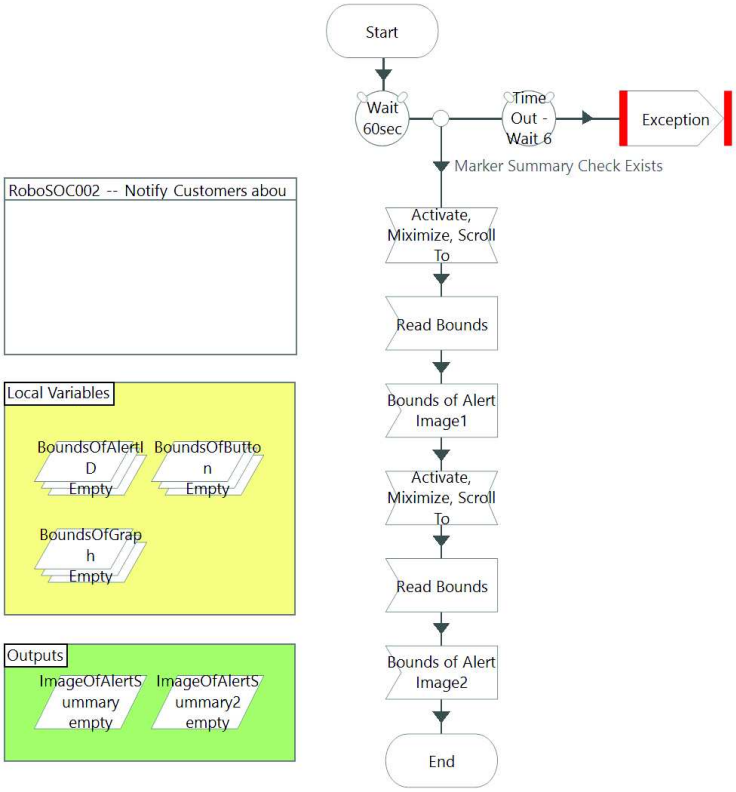
C.6 *Search Incident By ID* (*Object*)



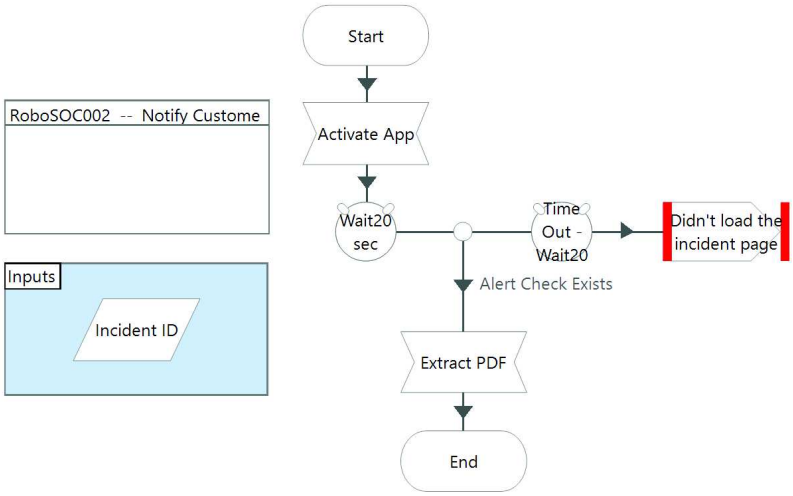
C.7 *Get Incident Details* (*Object*)



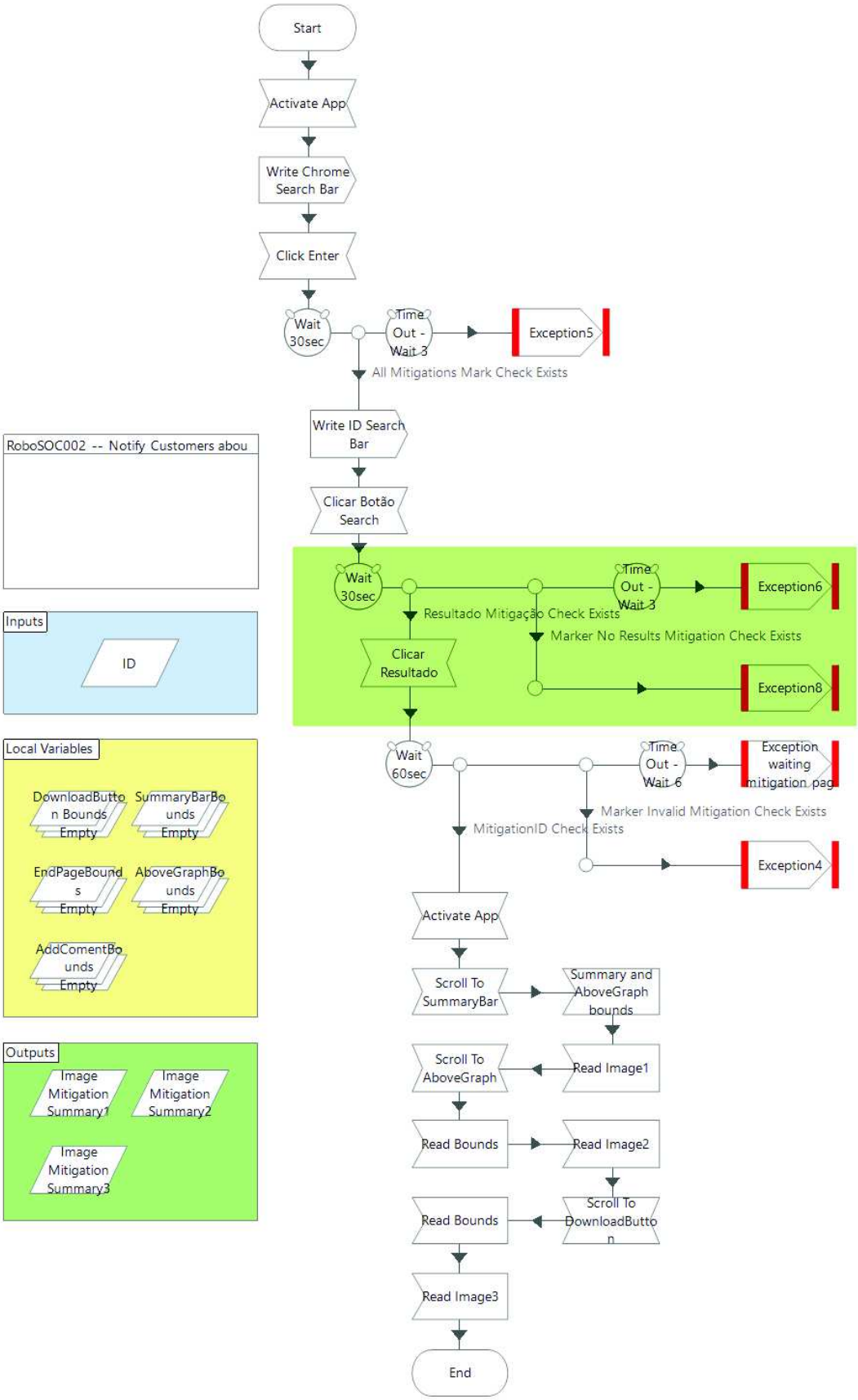
C.8 *Get Summary Screenshot* (*Object*)



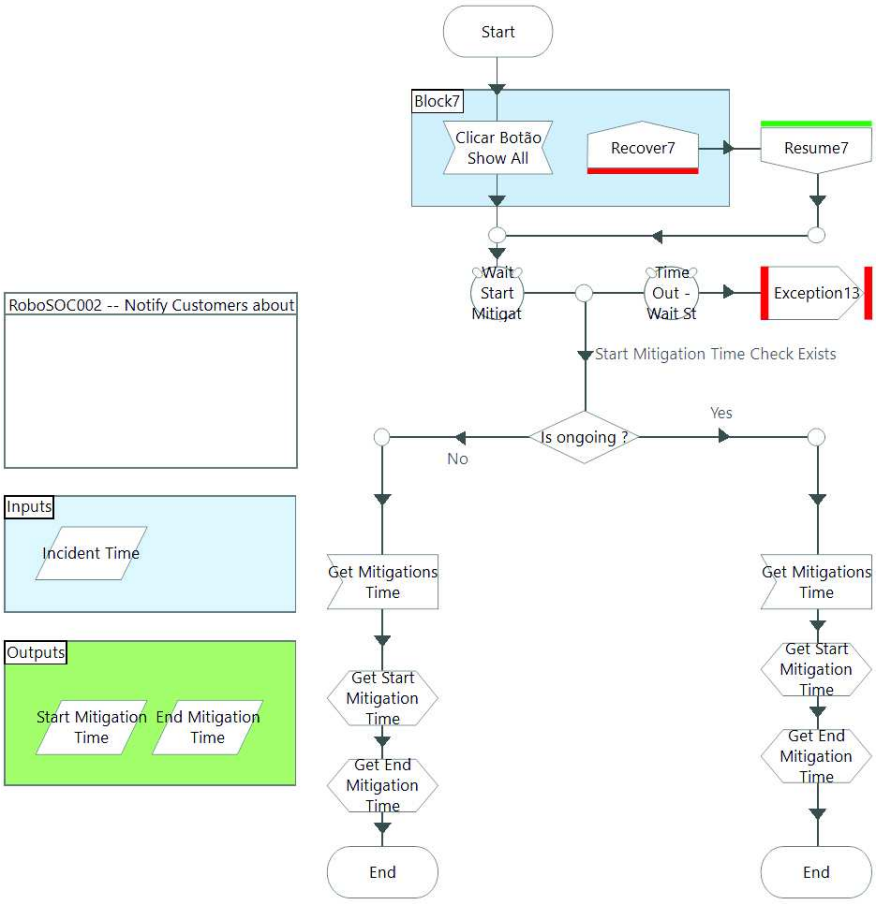
C.9 *Get Summary Report* (*Object*)



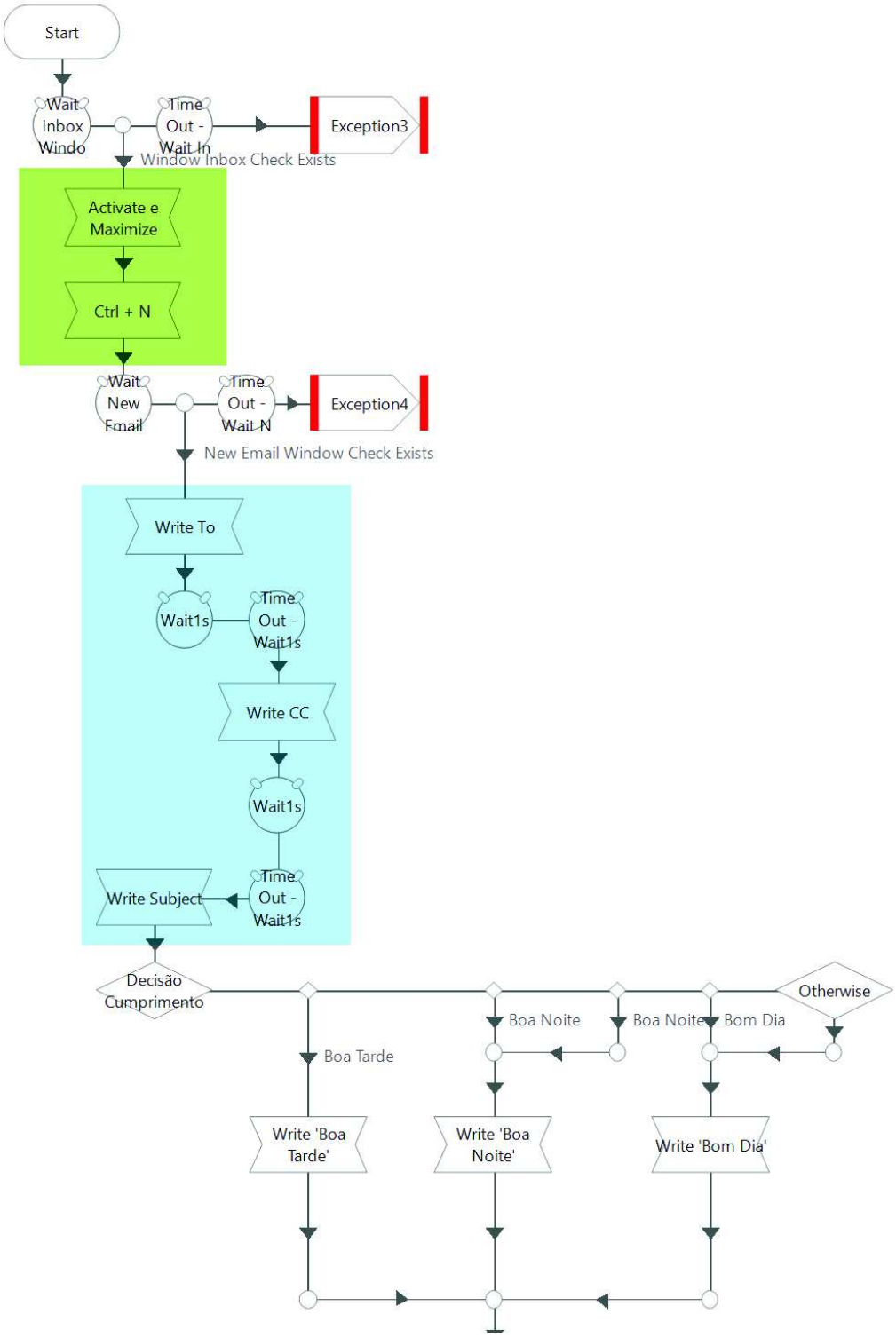
C.10 *Get Mitigation Screenshot* (*Object*)



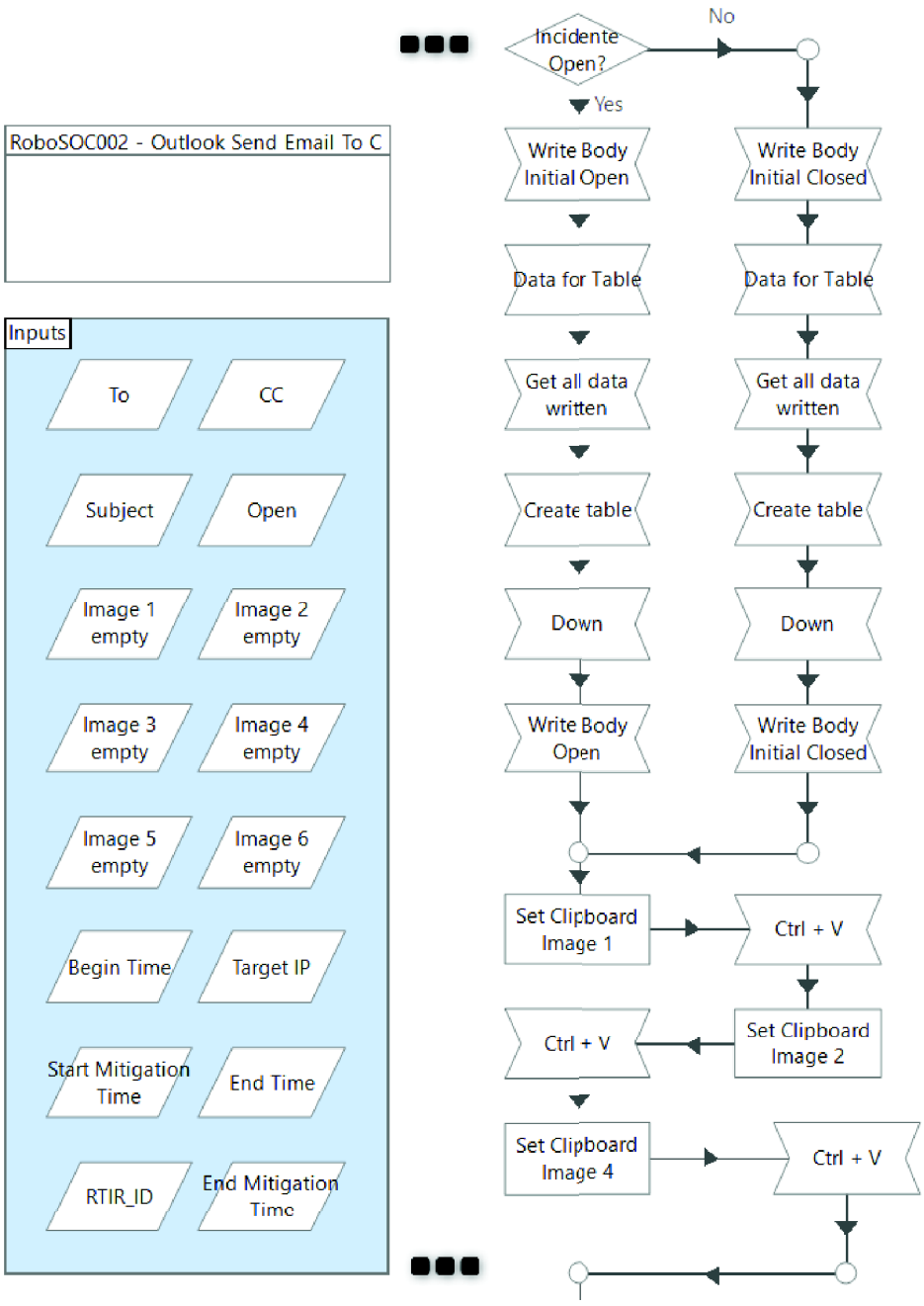
C.11 *Get Mitigation Times* (*Object*)



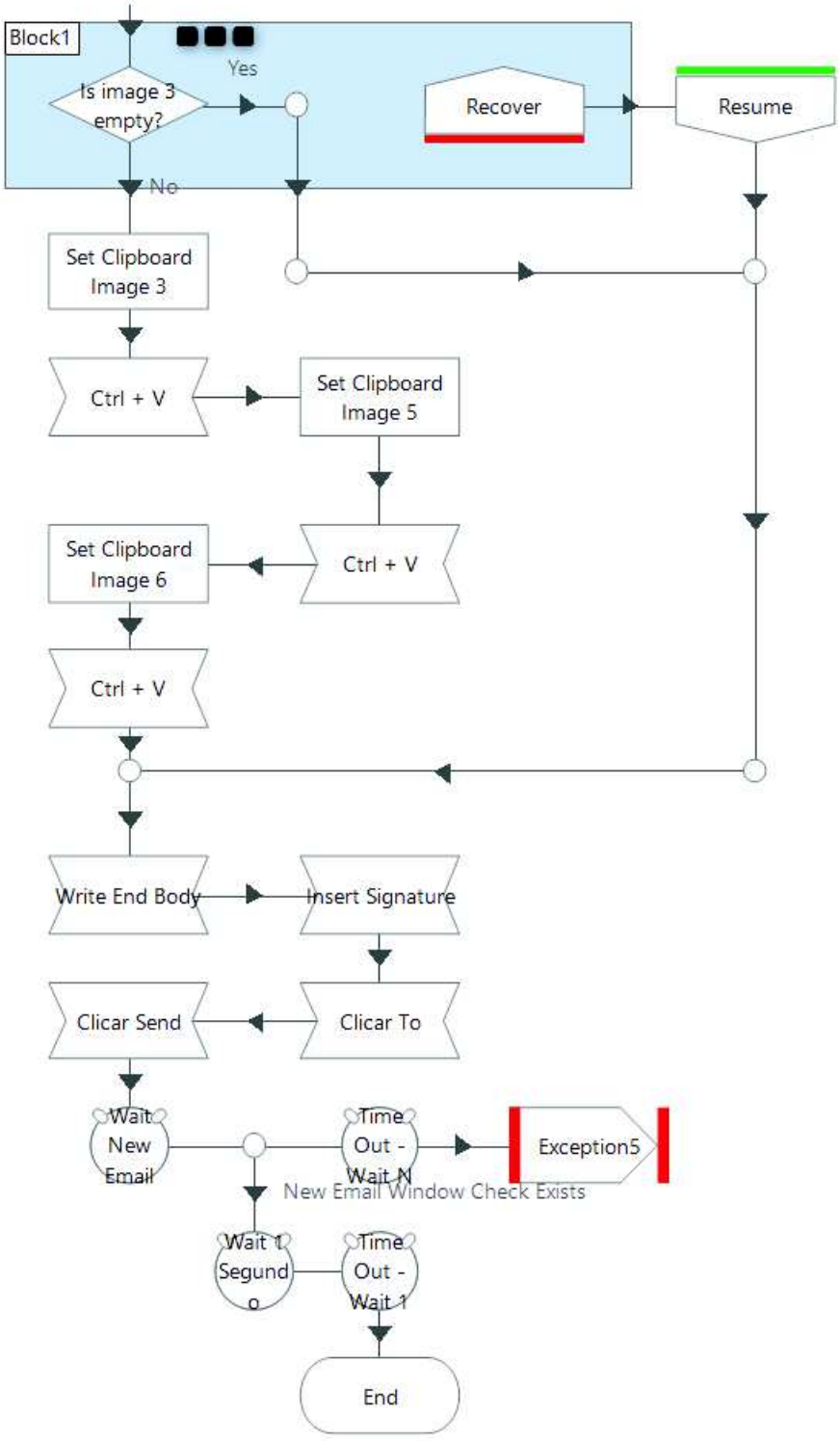
C.12 *Create E-mail part1* (*Object*)



C.13 *Create E-mail part2* (*Object*)



C.14 *Create E-mail part3* (*Object*)



Apêndice D

D.1 Tabela de conversão de datas de início e término do alerta

Input	*Output*	**Observações**
MMM dd HH:mm - Ongoing	yyyy-MM-dd HH:mm:ss	O alerta ainda não terminou.
MMM dd HH:mm – HH:mm	yyyyy-MM-dd HH:mm:ss yyyy-MM-dd HH:mm:ss	O alerta iniciou e terminou no mesmo dia e são devolvidas as datas de início e fim.
MMM dd HH:mm – MMM dd HH:mm	yyyy-MM-dd HH:mm:ss yyyy-MM-dd HH:mm:ss	O alerta iniciou e terminou em dias diferentes e são devolvidas as datas de início e fim.
MMM dd HH:mm yyyy - MMM dd HH:mm yyyy	yyyy-MM-dd HH:mm:ss yyyy-MM-dd HH:mm:ss	O alerta iniciou numa data de um ano e terminou noutro ano, ou um alerta que iniciou e terminou num ano já passado. São devolvidas as datas de início e fim.

D.2 Tabela de conversão de datas de início e término da mitigação

Input	*Output*	**Observações**
ddd MMM dd HH:mm:ss	yyyy-MM-dd HH:mm:ss	Esta estrutura serve tanto para a data de início como de término do tempo de mitigação de um alerta.