DOI: http://dx.doi.org/10.5769/C2011013

Uma arquitetura de software para extração de vestígios de programas de mensagens instantâneas

Vicente de Paula Melo Filho[1], Galileu Batista de Sousa[2], Gibeon Soares Aquino Jr.[3]
(1) vicente at peggasus.com.br, Peggasus
(2) galileu.gbs at dpf.gov.br, Departamento de Polícia Federal - Brasil
(3) gibeon at dimap.ufrn.br, Departamento de Informática e Matemática Aplicada - UFRN

Resumo* — *É cada vez mais comum o uso de programas de mensagens instantâneas para a comunicação entre as pessoas. Não obstante, eles podem ser utilizados para a realização de atividades ilícitas, tais como pedofilia, vazamento de informações corporativas confidenciais, ou qualquer outro crime digital. Devido a essas atividades, muitas vezes é necessário que se faça uma investigação criminal para capturar vestígios deixados por essas ferramentas. Para que essa captura seja feita de forma eficaz, pode ser necessária a utilização de ferramentas de software que consigam extrair e apresentar esses vestígios. Apesar da diversidade de ferramentas de mensagens instantâneas elas possuem praticamente o mesmo conjunto de funcionalidades e requisitos, de forma que é possível conceber uma metodologia para a captura e apresentação dos seus vestígios. Este trabalho define uma arquitetura de referência para a construção de ferramentas de software que vão capturar vestígios deixados por comunicadores de mensagens instantâneas. A proposta da arquitetura é ser extensível, promover o reuso de parte dos seus módulos e ser utilizada para capturar vestígios de mais de uma ferramenta de mensagens instantâneas. Para validar essa arquitetura, foi implementado um protótipo de um programa de captura de vestígios que são deixados pelo Skype 5.0.0.156 e pelo Windows Live Messenger 2009.

Palavras-chave — **Arquitetura de software; Arquitetura de Referência; Computação Forense; Ferramentas de Mensagens Instantâneas.**

Abstract* — *Nowadays it is popular to use Internet instant messengers for personal communications. Also, they are used for illicit activities, such as pedophilia, leaking confidential corporate information, any other digital crimes. Due to these activities, it is often necessary to make a criminal investigation to capture traces left by these tools. To do that effective, it is important to use software tools that can extract and present these traces. Despite the variety of messengers, they share virtually the same set of features and requirements, so it is possible to devise a methodology for the capture their traces. This work define a reference architecture for building software tools that will capture traces left by communicators for instant messaging. The proposed architecture is extensible, to promote the reuse of part of its modules and can be used to capture traces of more than one instant messaging tool. To validate this architecture, we built a prototype to capture program traces left by Skype 5.0.0.156 and Windows Live Messenger 2009.

Keywords — **Software Architecture; Reference Architecture; Computer Forensics; Instant Messaging Tools.**

1. Introdução

É crescente o número de indivíduos que utilizam ferramentas de mensagens instantâneas (*Instant Messengers* - IM), tais como Microsoft Windows Live Messenger (WLM) e Skype para interagirem com outras pessoas. Apesar dos benefícios, tais ferramentas podem ser usadas para fins ilícitos. Essas práticas podem deixar vestígios possíveis de serem encontrados nos diversos dispositivos em que ferramentas de IM são utilizadas.

De acordo com [1] as principais ferramentas de análise forense do mercado não oferecem os métodos apropriados de extração de listas de contatos, conversações ou dados trocados entre participantes de uma conversação.

Além disso, devido ao diversificado número de ferramentas de IM, à quantidade crescente de novas versões disponibilizadas e ao conjunto dos tipos de vestígios que podem ser deixados por esse tipo de ferramenta, devem-se adotar boas práticas da arquitetura de software para evitar que sejam criadas ferramentas de difícil extensão, manutenção, atualização e integração.

Este trabalho tem como objetivo propor uma arquitetura de software que sirva de referência para a construção de ferramentas que tenham como função a captura e apresentação de vestígios deixados por IMs. Objetiva-se que essa arquitetura sirva de diretriz, informando o que deve ser feito por essas ferramentas, elencando algumas funções que são obrigatórias e outras que são opcionais, permitindo que se desenvolva módulos de captura para novas ferramentas de IM que surjam no decorrer do tempo. Essa extensibilidade deve ser concebida com o menor impacto possível na estrutura geral do sistema. Para alcançar essa meta, serão propostos mecanismos de reusabilidade em boa parte dos componentes da arquitetura e mecanismos de variabilidade e extensibilidade nos pontos de especificidades de cada IM.

Este artigo está organizado como segue. Na seção II serão feitas breves observações sobre as ferramentas de IM e na seção III é feita uma proposta de arquitetura de software que capture vestígios deixados pelas ferramentas de IM e que ao mesmo tempo seja extensível para suportar novos mecanismos de identificação, captura, análise e apresentação desses vestígios. A Seção IV apresenta as principais visões arquiteturais da arquitetura proposta e na seção V serão explicados os pontos

de variação da arquitetura, que a torna extensível para se adaptar às novas ferramentas de IM que podem surgir e até mesmo às ferramentas já existentes, mas que ainda não estão implementadas. Na Seção VI é apresentada uma ferramenta construída para capturar os vestígios deixados pelo Skype 5.0.0.156 e pelo Windows Live Messenger 2009 e que utiliza as diretrizes da arquitetura de referência proposta. Por fim, na Seção VII os trabalhos futuros serão apresentados.

2. Ferramentas de Mensagens Instantâneas

Há milhões de usuários de ferramentas de IM no Brasil [9]. Ao redor do mundo, estima-se que o número de usuários que utilizam ferramentas de IM ultrapassa dois bilhões. Percebe-se na Tabela 1 que as ferramentas de IM utilizadas no contexto global são bastante diversificadas. Essa diversidade de serviços de IM dificulta o trabalho de construção de software para capturar os seus vestígios, pois será necessário construir programas de captura de vestígios para as mais variadas ferramentas. Essa pluralidade de comunicadores dificulta o desenvolvimento de ferramentas de captura de vestígios, pois pode ser custoso criar diversas ferramentas para atender a todas essas possibilidades.

De acordo com [2] apesar de existirem diferentes tipos de ferramentas de IM, as funcionalidades oferecidas são praticamente iguais, tais como: lista de contatos, troca de mensagens instantâneas, envio de mensagens off-line, envio de arquivos, *emoticons*, conferências por voz, conferências por vídeo, envio de e-mails, envio de mensagens SMS, *chat* multi-usuário e compartilhamento de arquivos.

Devido a essa similaridade de funcionalidades entre as ferramentas de IM, os tipos de vestígios produzidos por esse tipo de software possuem semelhança semântica. Assim sendo, surge a oportunidade de se criar uma padronização de captura, armazenamento e processamento desses dados.

3. A arquitetura de referência proposta

A arquitetura proposta nesse artigo foi inspirada nos trabalhos de [1] e [3], que descrevem implementações de ferramentas para capturar vestígios produzidos pelo WLM 8.0 e WLM 2009 respectivamente. Ideias importantes e que também tiveram influência na arquitetura foram adquiridas em [4], que descreve a automatização do processo de análise forense.

A. Requisitos para software de captura de vestígios

A análise das ferramentas WMM [1] e WMM 2009 [2] evidencia que as suas principais tarefas são:

- Identificação da localização dos vestígios;
- Extração dos vestígios (contatos e conversações);
- Integração dos vestígios;
- Apresentação dos vestígios.

Se pensarmos em termos de divisão de responsabilidades, já encontramos quatro funções desempenhadas por módulos específicos de um sistema de extração de vestígios de ferramentas de IM. Além disso, pode-se encontrar em [4] :

Uma parte do conhecimento do investigador relacionado à etapa de análise das evidências encontradas, que envolve uma série de raciocínios e tomadas de decisões, pode ser transferida para um sistema automatizado de análise forense.

O investigador deve ser capaz de configurar cada etapa do processo de investigação, de acordo com suas experiências de investigações anteriores.

Dessa forma já reconhecemos mais duas responsabilidades:

A. Configuração do software que irá fazer o processo de análise forense;

Análise dos vestígios encontrados através de técnicas data mining ou de inteligência artificial, por exemplo.

Tabela 1 - Número de usuários de algumas ferramentas de IM

Serviço	Número de usuários	Data
Gadu-Gadu	Mais de 15 milhões no total Mais de 6,5 milhões ativos (maioria na Polônia)	Fev/10
IBM Lotus Sametime	40 milhões no total	Dez/09
ICQ	50 milhões ativos (especialmente na Alemanha, Rússia, Leste Europeu e Israel)	Fev/10
Skype	23 milhões online (pico)	Out/10
	521 milhões no total	Set/10
Tencent QQ	100 milhões online (pico) (maioria na China)	Dez/10
	600 milhões de contas ativas (maioria na China)	Dez/10
	990 milhões de contas registradas no total. (maioria na China)	Out/09
VZOchat	1,1 milhão de usuários	Jan/10
Windows Live Messenger	330 milhões de usuários ativos	Jun/09
Xfire	16 milhões no total	Mai/10
Yahoo! Messenger	248 milhões de usuários ativos registrados no Yahoo (não se refere a todos os usuários do Yahoo Instant Messaging)	Jan/08
Facebook	Mais de 500 milhões de usuários ativos	Dez/10

Com isso encontramos seis principais responsabilidades que fazem parte do escopo de um software que automatiza o processo da captura de vestígios deixados por ferramentas de IM. Diante desses requisitos propomos um modelo de

processo que os satisfaça, obedecendo a uma sequência lógica. Nesse modelo de processo os seguintes requisitos funcionais (RF) devem ser contemplados:

1. **RF 01 – Configuração do processo**: O objetivo é usar o conhecimento do investigador para estabelecer parâmetros sobre as etapas que serão realizadas. Por exemplo, pode ser configurado:
   - A identificação automática dos vestígios, ou se o investigador informará a localização destes;
   - O capturador de vestígios a ser utilizado (entre os implementados). Ex.: Tecent QQ, WLM 2009, Yahoo Messenger;
   - O local de armazenamento dos resultados produzidos pelo sistema;
   - As análises (entre as disponíveis) a serem aplicadas sobre os vestígios;
2. **RF 02 – Identificação dos vestígios**: Tem como função identificar se existem vestígios na unidade investigada, descobrir qual o tipo de vestígio, identificar qual ferramenta o gerou e qual a sua localização;
3. **RF 03 – Captura dos vestígios**: Possui a função de extrair os vestígios que foram identificados e (eventualmente) decifrá-los;
4. **RF 04 – Integração dos vestígios**: O objetivo é integrar os vestígios que foram capturados e decifrados. Essa integração compreende:
   - Relacionar as entidades que representam os vestígios capturados. Por exemplo: os contatos com seus grupos de contatos e as conversações com os devidos contatos;
   - Unir todos os tipos de conversações em ordem cronológica. Por exemplo: o Skype armazena em diferentes locais os seus eventos, tais como trocas de mensagens de chat, chamadas de áudio, chamadas de vídeo e troca de arquivo. Há a necessidade de integrar esses diferentes vestígios em uma ordem cronológica para seja possível o investigador identificar qual o real conteúdo da interação entre os participantes de uma conversação;
5. **RF 05 – Análise dos vestígios**: Seu objetivo é ajudar o investigador com uma análise automatizada dos vestígios que foram capturados. Um tipo de análise que pode ser realizada é detectar se os arquivos trocados (e ainda encontrados durante a captura de vestígios) referem-se a fotos contém nudez humana [10].
6. **RF 06 – Apresentação dos vestígios**: A meta dessa atividade é apresentar os vestígios que foram integrados em um formato que tenha valor semântico para o investigador. Os vestígios podem ser exibidos em uma interface gráfica ou salvos em arquivos no formato XML, HTML ou texto;

Além dos requisitos e trabalhos já citados, [2] e [5] fornecem mais alguns requisitos importantes (alguns não-funcionais: RNF) a serem contemplados em uma ferramenta de captura de vestígios deixados por IM. São elas:

1. **RNF 01 – Precisão na captura dos vestígios**: O trabalho de captura de vestígios deve ser completado da forma mais eficiente possível, mas sem sacrificar a precisão;
2. **RNF 02 – Suportar mais de uma ferramenta de IM**: É importante que possam ser capturados vestígios de mais de um tipo de ferramenta de IM;
3. **RF 07 – Usar carving para capturar vestígios**: Alguns vestígios podem ser capturados através do processo de *data carving* [12];
4. **RF 08 – Armazenar os resultados em uma unidade diferente da que está sendo investigada**: O processo de captura de vestígios será realizado em uma unidade "somente-leitura". Os resultados devem ser armazenados em outra unidade;
5. **RF 09 – Armazenar os vestígios capturados em um repositório**: Os vestígios capturados e também aqueles que já foram integrados e analisados podem ser salvos em um repositório para serem reutilizados em etapas subsequentes do processo;
6. **RF 10 – Exibir os vestígios capturados em uma interface gráfica com o usuário**: Os vestígios que foram capturados e estão armazenados no repositório devem ser exibidos em uma interface gráfica para que o investigador possa analisá-los.

Outros requisitos também foram propostos para direcionar a arquitetura para um modelo de reusabilidade e extensibilidade, tais como:

7. **RNF 03 – Possibilidade de extensão:** Alguns módulos da arquitetura devem possibilitar a extensão de suas funcionalidades sem que seja necessário afetar a estrutura central do sistema desenvolvido**;**

Com base nas seis principais responsabilidades e nos demais requisitos identificados é que será proposta uma arquitetura de software que possa atender ao processo de captura de vestígios de ferramentas de IM. Como essa é uma proposta de arquitetura de referência, será dada ênfase ao que é necessário e não como se deve implementar os componentes que serão apresentados.

B. Módulos da arquitetura proposta

A arquitetura de referência possui seis módulos principais, cujas responsabilidades podem ser descritas da seguinte forma:

1. **Controle**: esse módulo é responsável por receber os comandos advindos do usuário que está utilizando o software, além de servir de entrada para as opções de configuração que serão usadas em etapas posteriores do processamento. É nesse módulo, por exemplo, que o usuário informa se quer que a ferramenta descubra

os vestígios ou se prefere informar quais vestígios procurar e onde eles estão.

2. **Identificação de vestígios**: esse módulo possui a função de identificar quais as ferramentas de mensagens instantâneas estão instaladas no computador, identificando suas respectivas versões e quais os tipos de vestígios encontrados, tais como: histórico de conversações, catálogo de contatos, etc. Alguns vestígios podem estar presentes no computador alvo até mesmo se o software que o gerou não está mais instalado. Até mesmo nesses casos, o módulo deve identificar esses vestígios.
3. **Captura de vestígios**: Esse módulo possui a responsabilidade de capturar os vestígios que foram identificados pelo módulo de identificação de vestígios e decifrar as informações, se necessário. A arquitetura também deve prover a possibilidade do usuário definir explicitamente qual vestígio ele deseja capturar independentemente do resultado alcançado pelo módulo de identificação de vestígios. Esses vestígios devem ficar armazenados no módulo de repositório de vestígios.
4. **Integração de vestígios**: A função desse módulo é integrar e relacionar os vestígios que foram capturados pelo módulo de captura de vestígios para que eles possam ser apresentados de uma forma mais clara ao usuário final e também para que possam ser utilizados pelo módulo de análise de vestígios. Os vestígios integrados também devem ser armazenados no repositório de vestígios.
5. **Análise de Vestígios**: A função desse módulo é analisar os dados que estão armazenados no repositório de vestígios e inferir algumas informações que podem ser relevantes para a descoberta de ações, comportamentos e relacionamentos do utilizador da ferramenta de mensagem instantânea. Uma vez descobertas, essas informações devem ser exibidas ao usuário.
6. **Apresentação de vestígios**: Esse módulo possui a responsabilidade de mostrar ao usuário final quais os vestígios encontrados. Essa exibição é feita utilizando os vestígios que estão no repositório de vestígios. A apresentação pode ser feita em interface gráfica, mas opcionalmente pode-se de exportar os vestígios em arquivos do tipo texto, XML ou HTML, por exemplo.

Além dos módulos acima, a arquitetura define um repositório de vestígios que tem a função de armazenar os principais resultados que são processados pelos módulos que capturam, integram e analisam os vestígios. Apesar desse repositório não ser um módulo da arquitetura, ele possui bastante relevância, pois servirá como canal de comunicação entre os módulos do sistema. O resultado de um módulo é armazenado no repositório e serve de entrada para outro módulo.

Pode ser visto em [9] o mapeamento entre os requisitos e os módulos da arquitetura de referência proposta.

4. Visões arquiteturais da arquitetura de referência

A arquitetura de referência proposta será exibida em duas das visões, são elas: visões de módulos e visões de componentes-e-conectores (C&C). Em cada uma destas visões serão apresentados um ou mais estilos, para complementarmente caracterizar a arquitetura.

A. Visão de Módulos

A visão de módulos, de uma forma geral, mostra como o software é decomposto em módulos e quais são os relacionamentos entre eles.

Para essa visão, serão apresentados os estilos de decomposição e de uso, contendo dois diagramas: um de mais alto nível, que apresenta apenas os módulos mais altos na hierarquia da arquitetura e outro diagrama mais detalhado, mostrando os submódulos e seus componentes de software.

- Estilo de decomposição

Para esse estilo, o critério de decomposição utilizado foi o princípio da divisão de responsabilidades. Cada módulo possui uma função bem definida e são decompostos em submódulos, que serão decompostos, e assim sucessivamente, até a granularidade ser fina o suficiente para ser viável sua implementação.

A Figura 1 mostra que uma decomposição de primeira ordem produziu seis módulos e um repositório (em notação UML [11]). Os módulos apresentados nesta figura foram ainda decompostos em submódulos conforme mostra a Figura 2. A descrição completa de cada um desses módulos e componentes pode ser encontrada em [9].

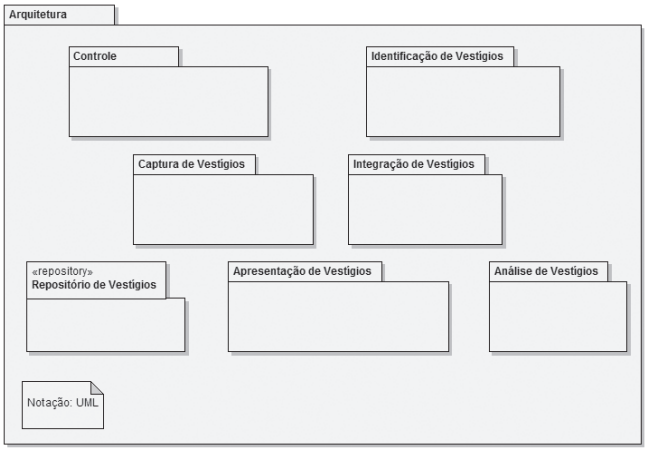

Figura 1 - Visão de decomposição em alto nível da arquitetura proposta

- Estilo "usa"

O estilo "usa" é utilizado para mostrar as relações de dependências entre os diversos módulos de um sistema.

A Figura 3 contém o diagrama do estilo "usa" da arquitetura. Esse diagrama mostra quais são as relações de dependência entre os módulos de nível mais alto da arquitetura proposta, onde cada módulo possua sua responsabilidade bem definida

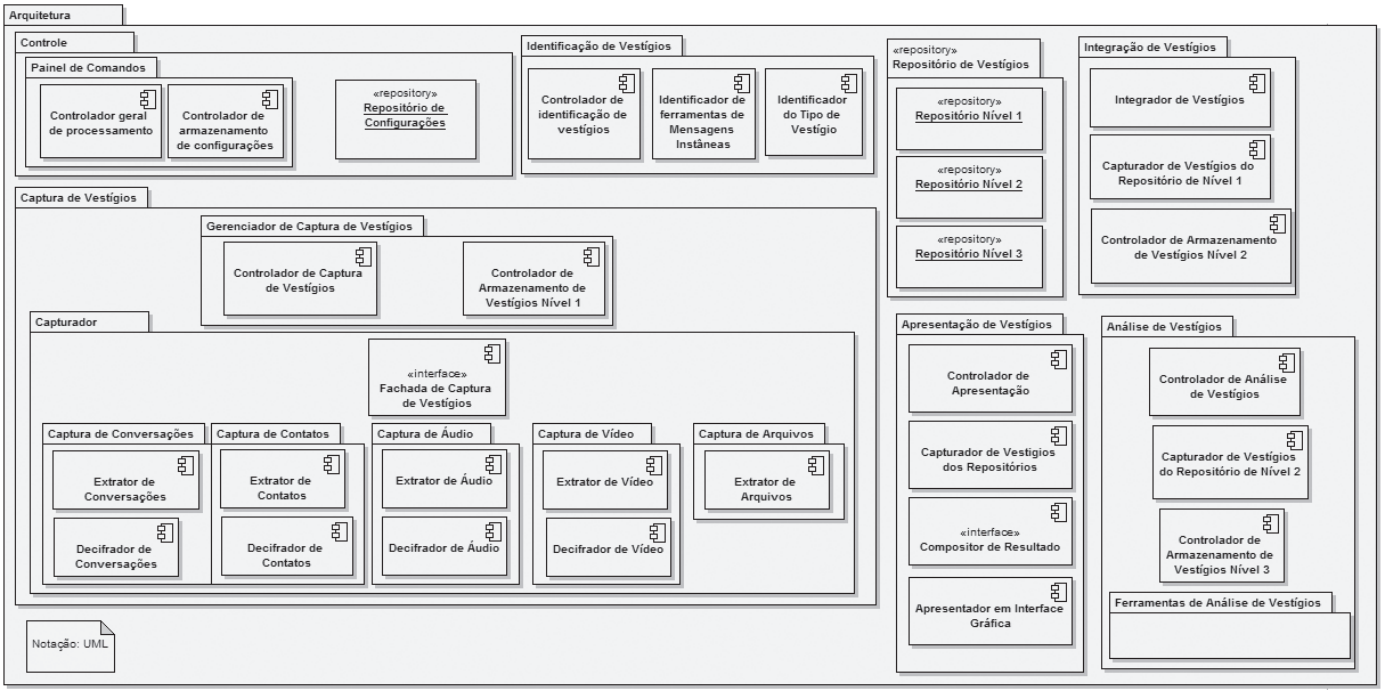

Figura 2 - Visão de decomposição refinada da arquitetura de referência proposta

e não influencia no comportamento interno de outro módulo, mas a dependência entre eles é inevitável, pois cada um faz apenas uma parte do processo completo.

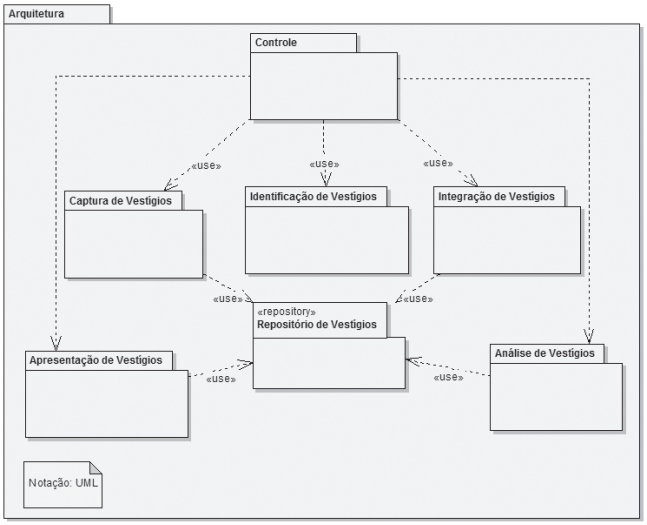

Figura 3 - Visão "usa" dos módulos de mais alto nível da arquitetura

Para a compreensão em maiores detalhes das dependências funcionais dos módulos, submódulos e componentes da arquitetura, é apresentada a Figura 4, que contém um diagrama mais minucioso do estilo "usa". Nesse novo diagrama as dependências entre cada um dos elementos da arquitetura fica visível e dessa forma tem-se uma visão bem mais precisa de como são as dependências entre cada um dos seus elementos de software.

A descrição completa de cada um dessas dependências funcionais pode ser encontrada em [9].

C. Visão componente-e-conector (C&C)

A visão C&C descreve os aspectos comportamentais da arquitetura e os seus elementos são componentes em tempo de execução e conectores [6], [7] e [8].

Na arquitetura proposta serão evidenciados três tipos diferentes de conectores:

1. ***Pipe-and-filter*:** É caracterizado por sucessivas transformações do fluxo de dados. Os dados chegam às portas de entrada de um filtro, são transformados e depois são passados, através de suas portas de saídas e utilizando o conector pipe, para o próximo filtro.
2. ***Call-return*:** Incorpora um modelo computacional em que componentes fornecem um conjunto de serviços que podem ser invocados por outros componentes. Um componente que invoca o serviço pausa (ou fica bloqueado) até que o serviço tenha sido concluído. Assim, é o equivalente arquitetônico de uma chamada de procedimento em linguagens de programação. Os conectores são responsáveis por transmitir a solicitação de serviço do requerente para o provedor e pelo regresso de todos os resultados.
3. ***Repository*:** Contem um ou mais componentes, chamados de repositórios, que normalmente mantêm grandes coleções de dados persistentes. Outros componentes podem ler e gravar dados dos repositórios. Em vários casos, o acesso a um repositório é mediada por um sistema de gerenciamento de banco de dados (SGBD) que fornece uma interface *call-return* para a recuperação e manipulação de dados.

- Estilo pipe-and-filter

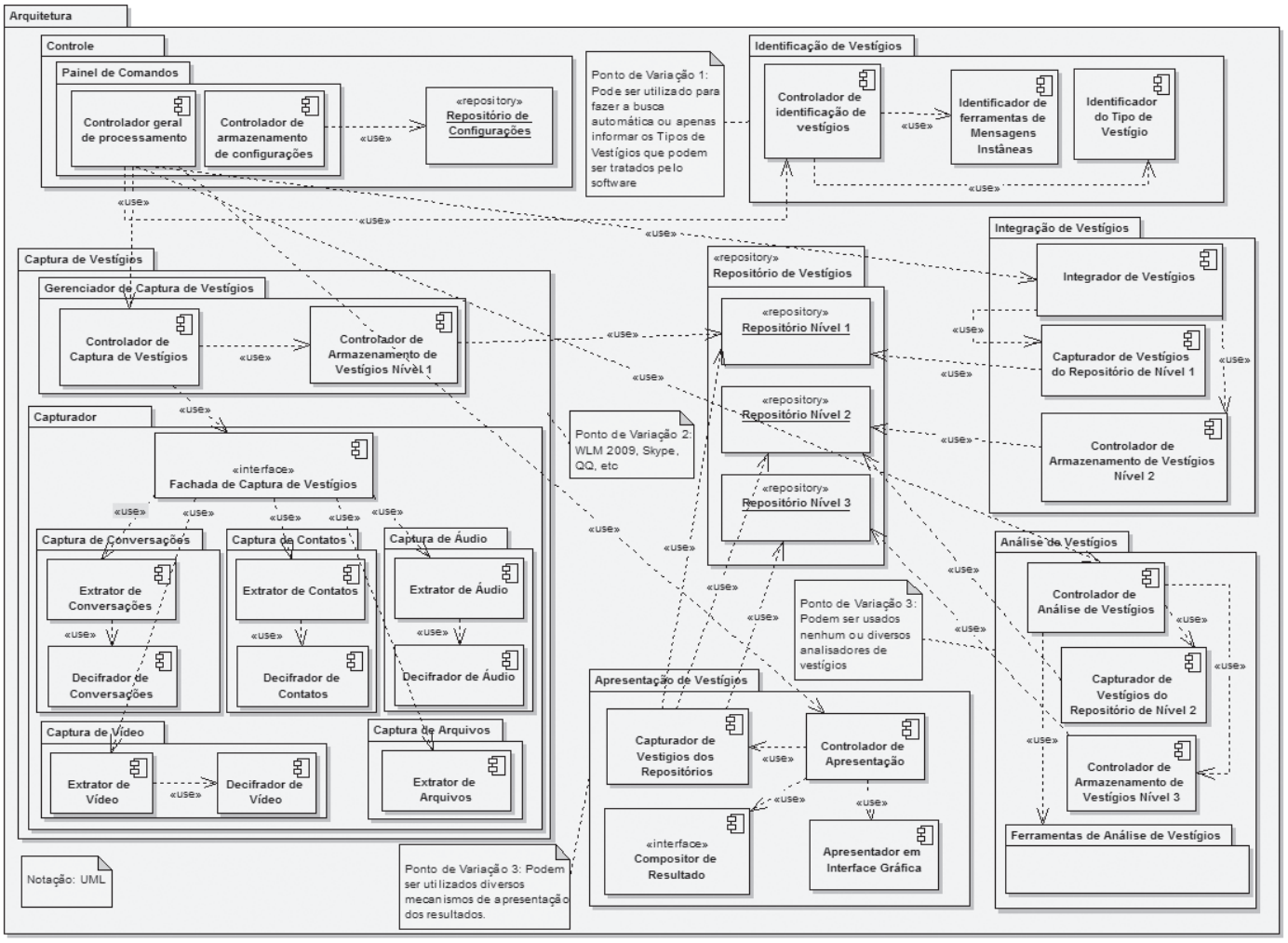

Figura 4 - Visão "usa" da arquitetura

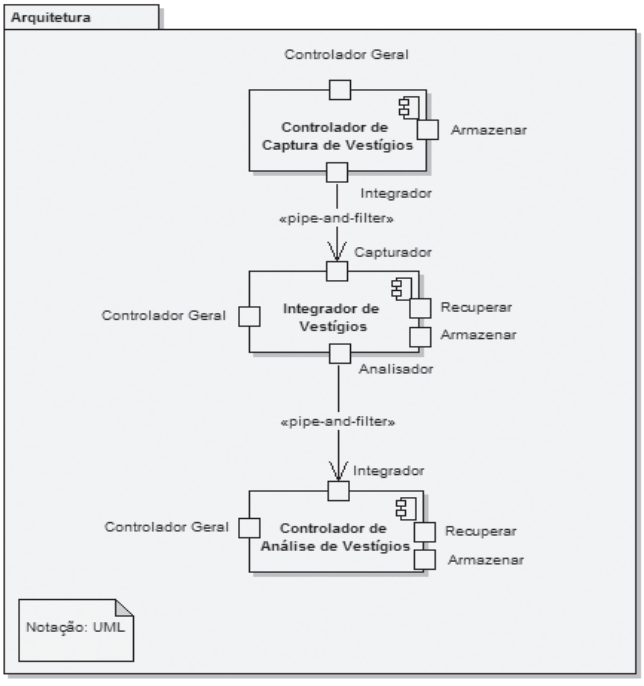

Figura 5 - Componentes que compõem o estilo pipe-and-filter da arquitetura proposta

A Figura 5 mostra os componentes que fazem parte do estilo *pipe-and-filter*. As funções de cada um desses componentes nesse estilo são:

1. **Controlador de Captura de Vestígios**: Captura os vestígios que foram encontrados, transforma-os na representação interna padrão e envia o resultado desse processamento, através da sua porta Integrador, para a porta Capturador do componente Integrador de Vestígios.
2. **Integrador de Vestígios**: Recebe os vestígios que foram capturados, integra-os e em seguida envia o resultado desse processamento, através da sua porta Analisador, para a porta Integrador do componente Controlador de Análise de Vestígios.
3. **Controlador de Análise de Vestígios**: Recebe os vestígios já integrados e, se configurado, faz a análise desses dados.

- Estilo *call-return*

Esse estilo está sendo utilizado na arquitetura para que um componente solicite um serviço a outro componente. Maiores informações sobre esse estilo podem ser vistos em [9].

- Estilo Repository

A Figura 6 mostra como estão dispostos os componentes do estilo *repository* da arquitetura proposta. A arquitetura provê um repositório para o armazenamento das configurações do sistema e três repositórios para armazenamento dos vestígios que foram capturados e processados. Maiores informações sobre as funções da cada um desses componentes podem ser vistos em [9].

5. Pontos de variação da arquitetura

Os pontos de variação de uma arquitetura são os locais onde se define que uma determinada flexibilidade deve estar presente. A flexibilidade é alcançada através do uso intencional de decisões específicas que deixam a arquitetura aberta. Nessa seção serão descritos os quatro pontos de variação da arquitetura proposta.

Em um sistema baseado nessa arquitetura pode ser necessário dar suporte a uma ferramenta de IM que não estava prevista anteriormente. Para alcançar esse objetivo, os módulos de identificação de vestígios e o módulo de captura de vestígios precisarão ser estendidos para que seja possível atender a este requisito. É mandatório que ambos módulos sejam incrementados para que o objetivo seja alcançado, pois não é coerente identificar um vestígio e não ter um capturador correspondente. Não deve ser possível capturar um vestígio de uma ferramenta de IM que não foi identificada, pois o conjunto dessas ferramentas suportadas é repassado para o módulo de controle e somente de posse dessa informação que o controlador geral de processamento solicita ao módulo de captura que realize o processo de captura de vestígios de determinado IM que foi selecionado dentre os suportados.

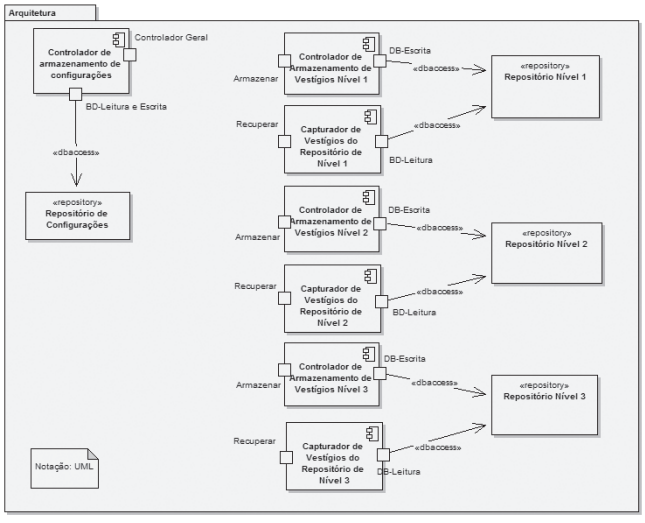

Figura 6 - Componentes que compõem o estilo repository da arquitetura proposta

A arquitetura de referência proposta não suporta apenas a evolução da arquitetura com a adição de novos mecanismos. Há também a possibilidade de substituição de um dos seus componentes de identificação, captura, análise e apresentação de vestígios por outro componente que faça a mesma operação com um desempenho melhor ou que corrija algum possível problema.

A. Variação no módulo de Identificação de Vestígios

Para que este módulo identifique os vestígios de uma ferramenta que não estava prevista, serão necessárias extensões nos componentes de identificação de ferramentas e de identificação de vestígios. Esses componentes em si não identificam ferramentas e nem vestígios. Ao invés disso, eles delegam essa tarefa para um dos identificadores específicos das ferramentas de IM que estão registrados no módulo. Por exemplo, para dar suporte à nova versão do Skype, será necessário criar um componente que identifique a presença do Skype e outro componente que identifique vestígios deixados por essa nova ferramenta de IM. O primeiro componente ficará registrado no identificador de ferramentas de IM e o segundo ficará registrado no identificador de vestígios. A partir desse registro o módulo pode usar esses novos componentes quando estiver fazendo identificação de ferramentas ou vestígios.

Dessa forma, o módulo possuirá um conjunto de pares de identificadores específicos (um identificador de ferramenta e um de vestígios) e poderá ser estendido para dar suporte a uma nova ferramenta de IM com o registro de um novo par de identificadores.

B. Variação no módulo de Captura de Vestígios

Para dar suporte a uma nova ferramenta de IM que não estava prevista, um novo submódulo capturador (e seus componentes) precisa ser implementado e registrado no controlador de captura de vestígios. Apenas a fachada desse módulo precisa ser registrada, mas os componentes de extração e decifração precisam ser implementados especificamente para a nova ferramenta de IM que será suportada. De posse desse conjunto de submódulos de captura que estão registrados, o controlador do módulo pode delegar a tarefa de capturar os vestígios para o submódulo competente.

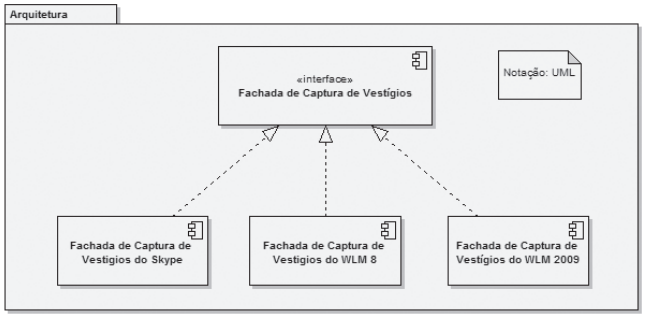

Figura 7 - Ponto de variação no módulo de Captura de Vestígios

Observa-se na Figura 7 que o elemento Fachada de Captura de Vestígios corresponde a uma interface para três diferentes tipos de capturadores de vestígios. Dessa forma, pode ser observado que a arquitetura dá suporte à captura de vestígios de múltiplas ferramentas de IM. Os possíveis capturadores das ferramentas de IM devem ser implementados de forma independente (e devem implementar a interface Fachada de Captura de Vestígios) e podem ser adicionados ao sistema como forma de estender as suas funcionalidades.

C. Variação no módulo de Análise de Vestígios

Para que seja adicionado ao sistema um novo mecanismo de análise de vestígios, deve ser criado um componente independente e este deve ser registrado no controlador do módulo de análise. O controlador então informa ao módulo de controle quais são os possíveis mecanismos da análise disponíveis. O investigador pode usar nenhum, um, ou vários dos mecanismos disponíveis durante o processo de extração de vestígios. Pode ocorrer a situação de não haver nenhum analisador disponível.

D. Variação no módulo de Apresentação de Vestígios

Esse módulo também provê extensibilidade à arquitetura, pois possui um mecanismo de registro de compositores de resultados, que funcionam de forma independente e utilizam os dados que estão armazenados nos repositórios de vestígios para realizar seu trabalho.

Para que seja adicionado ao sistema um novo mecanismo de composição de resultados deve ser criado um componente, que implemente a interface Compositor de Resultado, e registrado no Controlador de Apresentação. Este controlador informa ao módulo de Interface com o Usuário quais são os mecanismos de apresentação disponíveis para que o investigador possa selecioná-los.

6. Implementação de uma arquitetura

Nesta seção será demonstrada a implementação de uma ferramenta de captura de vestígios deixados por IMs que utiliza a arquitetura proposta. O software captura vestígios deixados pelas ferramentas Skype (versão 5.0.0.156) e WLM 2009. A escolha destas duas ferramentas deve-se ao fato delas serem bastante populares no Brasil.

A captura de vestígios do Skype 5.0.0.156 foi desenvolvida durante o período de elaboração da dissertação de mestrado [9] e obedece desde o início ao modelo aqui proposto. Por outro lado, a captura de vestígios do WLM 2009 foi adaptada do trabalho de [3]. Como a implementação desta última não foi concebida de acordo com o modelo de arquitetura proposto, foram necessários alguns *refactorings* no código fonte da ferramenta WMM 2009 para obedecer às diretrizes definidas na arquitetura. A linguagem de programação utilizada foi C# para facilitar o reaproveitamento do código fonte do WMM 2009.

A. Implementação dos módulos

Os seis módulos propostos na arquitetura foram implementados, mas alguns com pequenas restrições. São elas:

1. **Módulo de Identificação de Vestígios**: os componentes de identificação de ferramentas de IM e de vestígios de IM não foram implementados na sua totalidade. Eles apenas estão simulando o processamento e devolvendo uma lista de resultados já pré-definidos;
2. **Módulo de Captura de Vestígios**: A captura de áudio, vídeo e dos arquivos que foram transferidos durante conversações não foi implementada;
3. **Módulo de Análise de Vestígios**: Os mecanismos de análise criados servem para que se tenha uma noção da potencialidade das análises que podem ser feitas a partir dos vestígios capturados.
4. **Repositório de Vestígios**: Os vestígios não serão armazenados em um sistema gerenciador de banco de dados. Ao invés disso, estarão apenas na memória RAM do computador, enquanto o programa estiver sendo executado.

B. Módulo de Controle

Esse módulo serve para controlar as operações que serão realizadas pelo usuário, para configurar o sistema e controlar o processamento dos outros módulos. Ele não precisa ser alterado para dar suporte a uma nova ferramenta de IM. Quando for o caso, é necessário apenas adicionar os módulos que identificam e capturam os vestígios.

A Figura 8 apresenta a tela inicial do programa. Podem ser observadas as opções para executar cada uma das etapas do processo de captura de vestígios. Há também a opção para executar todas as etapas do processo e, por fim, a opção para definir as configurações do sistema.

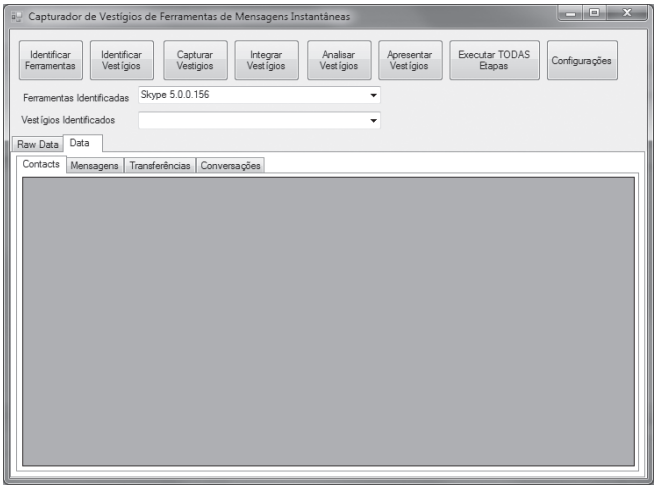

Figura 8 - Tela Inicial do software de captura de vestígios

C. Módulo de Identificação de vestígios

Esse módulo tem a função de identificar ferramentas de IM e os vestígios deixados por elas. O processo de identificação é feito por componentes que foram criados para dar suporte a uma determinada ferramenta e que, para isto, implementam as interfaces de identificação de ferramentas e de identificação de vestígios. Por exemplo, no caso da identificação de ferramentas e vestígios do Skype 5, foram criados os seguintes componentes: *IdentificadorDoSkype5* e *IdentificadorDeVestigiosDoSkype5*, que implementam respectivamente as interfaces *IPluginIdentificadorDeFerramentas* e *IPluginIdentificadorDeVestigios*.

Sempre que for necessário dar suporte a uma nova ferramenta de IM, este módulo precisará ser expandido. As modificações necessárias são:

1. Adicionar o componente específico de identificação da nova ferramenta que será suportada. Este componente deve implementar a interface *IPluginIdentificadorDeFerramentas*;
2. Adicionar o componente específico de identificação de vestígios da nova ferramenta que está sendo suportada. Este componente deve implementar a interface *IPluginIdentificadorDeVestigios*;
3. Registrar, no componente de identificação de ferramentas de IM, o novo componente que faz a identificação da nova ferramenta que será suportada;
4. Registrar, no componente de identificação de vestígios, o novo componente que faz a identificação de vestígios deixados pela nova ferramenta que será suportada.

Registrar um componente significa torná-lo um *plugin* para outro componente, geralmente chamado de hospedeiro, que reconhecerá quais são os *plugins* disponíveis para realizar determinada tarefa. Tal estrutura é uma boa estratégia para prover a extensibilidade da arquitetura, pois eles podem ser adicionados ou removidos sem causar impacto na sua estrutura.

Diversas abordagens podem ser utilizadas para prover o mecanismo de *plugins*, mas de uma forma geral, este tipo de componente implementa uma interface que o identifica como plug-in e o componente hospedeiro implementa outra interface que o caracteriza como hospedeiro. Os métodos oferecidos por essas duas interfaces serão utilizados na comunicação entre eles. Podem ser encontradas mais informações sobre esse módulo em [9].

D. Módulo de Captura de Vestígios

Esse módulo é o responsável por capturar, decifrar e armazenar, nos repositórios, os vestígios que foram deixados pelas ferramentas de IM. Para que esse processo ocorra devem ser informados ao Controlador de Captura de Vestígios os seguintes parâmetros:

1. **Ferramenta de IM e sua versão**: serve para delegar o processamento para um componente capturador específico de uma ferramenta de IM;
2. **Tipo de vestígio**: informa qual o tipo de vestígio que deve ser capturado, por exemplo: contatos, conversações, áudio, vídeo, etc;
3. **Localização do vestígio**: informa ao componente de captura onde se encontra o vestígio que será capturado.

O processo da captura é feito por componentes específicos da versão da ferramenta que gerou os vestígios. Esses componentes devem implementar a interface *IPluginFachadaDeCapturaDeVestigios* e se registrar no controlador de captura de vestígios, pois este é o hospedeiro de *plug-ins* de capturadores de vestígios.

Dependendo do parâmetro que informa a ferramenta de IM e a sua versão, o controlador de captura de vestígios irá delegar o processamento para determinado capturador de vestígios. Este, por sua vez, usará o parâmetro que informa o tipo de vestígio e delegará o processamento para o componente de captura específico, por exemplo: capturador de conversações, contatos, etc.

- Captura de vestígios do Skype 5.0.0.156

Os vestígios do Skype, na sua versão 5.0.0.156, ficam armazenados em um banco de dados no formato SQLite10. Nesse banco de dados há 17 tabelas, que possuem relacionamento entre si, e armazenam os vestígios que são deixados por essa ferramenta de IM. O conteúdo dessas tabelas não está cifrado e, portanto não foi necessário implementar os componentes que decifram os vestígios.

Foram implementados o componente de captura dos contatos, que utilizou as tabelas *Contacts* e *ContactsGroups*, e o componente de captura das conversações, que utilizou as tabelas *Calls*, *CallMembers*, *Chats*, *ChatMembers*, *Conversations*, *Messages*, *Participants*, *SMSes* e *Transfers*.

- Captura de vestígios do WLM 2009

Conforme [3], os contatos do WLM 2009 estão organizados sob a arquitetura do banco de dados ESENT. Este não possui drivers de acesso com interfaces ODBC ou JDBC, por isso é necessário utilizar a Esent Win32 API que está encapsulada em uma DLL (EsentInterop.dll) para extrair os contatos do WLM 2009. O WMM 2009 que fora produzido no trabalho [3] foi adaptado a essa arquitetura para que fosse possível capturar os vestígios referentes a contatos e conversações dessa ferramenta de IM.

E. Módulo de Integração de Vestígios

Esse módulo possui a função de integrar os vestígios que estão armazenados no repositório de vestígios. Ele não precisa ser modificado para que o sistema passe a fornecer suporte a uma nova ferramenta de IM, haja vista, que o formato dos dados armazenados nos repositórios de vestígios obedece a um padrão para todas as ferramentas que o sistema suporta.

F. Módulo de Análise de Vestígios

Esse módulo possui a função de realizar as análises dos vestígios que estão armazenados no repositório. Para que esse processo seja realizado devem ser informados ao controlador de análise de vestígios quais são os mecanismos de análises que serão utilizados. Como esse controlador é o hospedeiro dos componentes de análise ele implementa a interface *IPluginHostAnalisadorDeVestigios*.

A análise é feita por componentes específicos e que implementam a interface *IPluginAnalisadorDeVestigios* e que se registram no controlador de análises de vestígios. Não é necessário alterar esse módulo para dar suporte a uma nova ferramenta de IM, pois a arquitetura do sistema provê o acesso de maneira uniforme aos vestígios que estão armazenados no repositório.

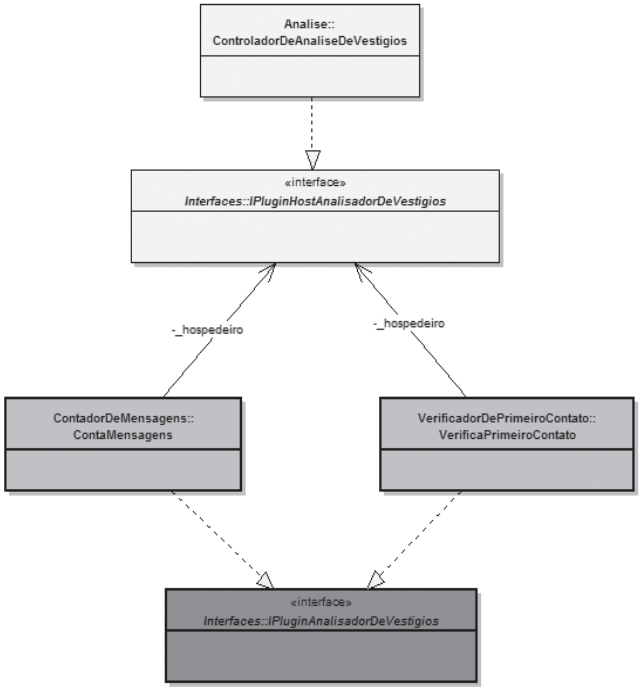

Figura 9 - Diagrama de classes do módulo de Análise de Vestígios

A Figura 9 mostra um diagrama com as principais classes e interfaces do módulo de análise de vestígios. Os componentes que executam a análise, tais como: *ContadorDeMensagens* e *VerificaPrimeiroContato* implementam a interface *IPluginAnalisadorDeVestigios*.

G. Módulo de Apresentação de Vestígios

A responsabilidade deste módulo é montar as apresentações dos vestígios que estão armazenados nos três níveis do repositório. Essa apresentação pode ser resumida como recuperar os vestígios do repositório e convertê-los para arquivos com layout e formatos específicos, tais como XML, HTML ou apresentá-los em uma interface gráfica. Para que esse processo seja realizado devem ser informados ao controlador de apresentação quais mecanismos de composição serão suportados. Como esse controlador é o hospedeiro dos componentes de apresentação ele implementa a interface *IPluginHostApresentadorDeVestigios*.

Este módulo não necessita modificação para suportar uma nova ferramenta de IM, pois seus componentes farão acesso ao repositório que é fornecido pela arquitetura do sistema. Se um novo mecanismo de análise de vestígios for adicionado ao sistema, este módulo também não precisará ser modificado, pois os resultados das análises também serão recuperados do repositório de vestígios.

7. Trabalhos futuros

Como o software que foi implementado ainda é um protótipo, não está pronto para ser usado em uma investigação criminal real. As atividades seguintes podem torná-lo uma ferramenta efetiva para computação forense:

A. Módulo de Controle:
   1. Criar mecanismos para consultar as investigações que já foram realizadas;
   2. Implementar novas formas de visualizar os vestígios na interface gráfica. Por exemplo, colocar mecanismos para filtrar os vestígios que foram capturados.

B. Módulo de identificação de Vestígios:
   1. Implementar os mecanismos de identificação de ferramentas e de identificação de vestígios. Atualmente essas duas funcionalidades foram simuladas no protótipo para as ferramentas Skype 5.0.0.156 e WLM 2009;

C. Módulo de captura de vestígios:
   1. Implementar o mecanismo de armazenamento de vestígios em um SGBD. Nesse protótipo, os vestígios estão apenas na memória RAM.

D. Módulo de análise de vestígios:
   1. Implementar mecanismos de análises que sejam relevantes em uma investigação criminal, por exemplo: quais arquivos trocados fazem apologia à pornografia infantil ou a outro crime.

E. Módulo de apresentação de vestígios:
   1. Implementar a apresentação dos vestígios em arquivos HTML com hiperlinks entre os dados correlatos, por exemplo: Ao se clicar em um contato, pode-se exibir quais as conversações que foram feitas com aquele contato.

Referências

[1] SOUSA, Galileu Batista. WMM - Uma ferramenta de extração de vestígios deixados pelo Windows Live Messenger. In: ICCYBER 2008, Rio de Janeiro: Anais, 2008. P. 104-111.

[2] NUNES, Gabriel Menezes. Instant Messaging Forensics. In: ICCYBER 2008, Rio de Janeiro: Anais, 2008. p. 18-30.

[3] MEDEIROS, M. H. F; SOUSA, G. B. Extração de vestígios do Windows Live Messenger 2009. In: ICCYBER 2009, Natal: Anais. Centro de Convenções, 2009. p. 15-20.

[4] ABDALLA, Marcelo. Forense computacional e sua aplicação em segurança imunológica. Campinas: UNICAMP, 2003. 241 p. Dissertação (Mestrado) - Instituto de Computação, Universidade Estadual de Campinas, Campinas, 2003.

[5] GAO, Yuhang; CAO, Tianjie. Memory Forensics for QQ from a Live System. In: School of Computer, China University of Mining and Technology. 2010, p. 541-548.

[6] BASS, Len; CLEMENTS, Paul; KAZMAN, Rick. Software Architect in Practice. Canadá: Addison Wesley, 2003.

[7] GORTON, Ian. Essential Software Architecture. Berlim: Springer, 2006.

[8] CLEMENTS, Paul, et al. Documenting software architectures: views and beyond. 2a ed. Addison Wesley, 2010.

[9] MELO FILHO, Vicente de Paula. Uma proposta de arquitetura de software para extração de vestígios de programas de mensagens instantâneas. Recife: CESAR, 2011. 141 p. Dissertação de Mestrado – Centro de Estudos e Sistemas Avançados de Recife, 2011.

[10] ELEUTERIO, Pedro; POLASTRO, Mateus. Optimization of automatic nudity detection in high-resolution images with the use of NuDetective Forensic Tool. In: ICCYBER 2010, Brasília: Anais, 2010. p. 59-66.

[11] LARMAN, Craig. Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Interative Development. 3a ed. Pearson Education, 2005.

[12] REYES, Anthony; et al. Cyber Crime Investigations: Bridging the Gaps Between, Security Professionals, Law Enforcement, and Prosecutors. Elsevier Science, 2007.