Faculdade de Engenharia da Universidade do Porto

Departamento de Engenharia Electrotécnica e de Computadores

Mestrado em Redes e Serviços de Comunicação

Monitorização Electrónica de Provas de Exame

Pedro Filipe Cruz Pinto

Licenciado em Engenharia Electrotécnica e de Computadores pela

Faculdade de Engenharia da Universidade do Porto

Dissertação submetida para satisfação dos requisitos para obtenção do grau de

Mestre em Redes e Serviços de Comunicação

Dissertação realizada sob a supervisão de

Professor Assistente Convidado João Isidro Araújo Vila Verde

Porto, Julho de 2007

Resumo

Hoje em dia já nos habituamos ao ritmo veloz da evolução tecnológica a que vamos assistindo. Os sistemas informáticos são alvo de avanços constantes e apresentam um potencial crescente, como o aumento de velocidades de processamento, a melhoria da qualidade em interfaces de utilizador e consequentemente maior facilidade de utilização, etc. Cada vez mais se recorre a estes sistemas para automatizar tarefas que tanto substituem por completo o papel do ser humano, como desempenham tarefas que exigem capacidades que este não possui e, consequentemente, nunca as poderia desempenhar. A monitorização de acções é um tipo de tarefa que pode ser automatizada com recurso às novas tecnologias de informação. Este é o conceito base deste documento que é, por sua vez, explorado no contexto de um ambiente de formação específico. Num processo de formação, a avaliação desempenha um papel fundamental. É na avaliação que os formandos justificam os seus conhecimentos, e é também nesta fase que se levantam algumas questões quanto à influência de alguns factores que não ajudam na avaliação correcta dos conteúdos avaliados. Existem vários factores relacionados com o aluno ou com o método de realização da prova que podem adulterar o resultado e consequentemente levar a uma incorrecta avaliação.

Neste documento é apresentado um método de monitorização dinâmica para a realização de provas electrónicas (provas realizadas por computador), que tem por objectivo aumentar o grau de eficácia na prevenção destes factores, fazendo com que a avaliação seja a mais correcta e mais justa possível. Baseado neste método de monitorização, foi especificado, desenvolvido e testado um protótipo, do qual são apresentados os resultados que foram obtidos.

Abstract

Nowadays, we are used to fast technological advances faced on daily basis. The computer-based systems are object of constant advances and they provide an increasing potential, with the increasing processing speeds, better quality user interfaces, more "user-friendly" and therefore, more easy to use and useful in some areas. Currently, task automating systems that substitute completely human-role, or substitute him on tasks demanding extra capacities that he does not possess (and therefore could never perform them), are applied more frequently. Action monitoring is a type of task that may be automated applying new information technologies. This is the main concept presented in this document, which is explored in a specific educational environment. In an educational process, the evaluation constitutes a fundamental role. It is in the evaluation process that students justify their knowledge, and it is also in this phase that some questions appears related to the influence of some factors that do not help to the correct evaluation of the evaluated contents. There may exists some factors related with the pupil or with the test method applied, which can influence the results, falsifying evaluation that will not serve it initial intentions.

In this document it is presented a dynamic monitoring method for electronic tests (tests carried through a computer system), to apply in a specific environment, which has the objective of increasing the degree of effectiveness in the prevention of these factors, turning the evaluation more precise and as fair as possible. Based on this monitoring method, a prototype has been specified, developed and tested, and its results are also presented.

Conteúdo

Lista de Tabelas e Código

Lista de Figuras

Agradecimentos

Este trabalho foi possível com a ajuda e motivação de várias pessoas que partilharam comigo as longas horas de trabalho. Desta forma, quero louvar a ajuda incansável do meu orientador, o Professor Isidro Vila Verde que, com os seus conselhos sábios, a sua paciência e dedicação, orientou e fez crescer este projecto.

Por fim, dedico este trabalho à minha família, em particular aos meus pais, irmã e avós que me apoiaram e ajudaram em todos os momentos, e de forma especial à Xana que percorreu este caminho comigo.

Capítulo 1 - Introdução

O domínio dos sistemas de informação abre os horizontes para novas aplicações em diversas áreas de interesse. Consequentemente, novos sistemas podem ser projectados, construídos e utilizados como ferramentas úteis em vários contextos, cujos objectivos podem passar, por exemplo, pela optimização de resultados, pela obtenção de menores custos e/ou maiores proveitos, obtenção de melhores desempenhos, regular actividades ilícitas, entre outros. Em qualquer destes casos, pode dar-se a substituição total ou substituição parcial da figura humana. No caso da substituição total, podem ser observados efeitos de ordem social, fruto da substituição em massa dos recursos humanos e, em qualquer dos casos apresentados, podem ser levantadas questões de segurança (privacidade e autenticidade) que aumentam a dúvida acerca da eficácia e transparência de alguns processos realizados por computador. Apesar destas questões serem relevantes, não são objecto deste documento, merecendo apenas uma breve referência.

Uma das áreas da aplicação dos sistemas de informação é a área da formação. O processo de formar ou educar, implica a troca de informação entre indivíduos sob várias formas, que promovem a aprendizagem e o conhecimento [1]. Várias fases podem ser distinguidas na aprendizagem, como a aquisição, a consolidação e avaliação do conhecimento. Em todas estas fases de aprendizagem, os sistemas de informação são cada vez mais utilizados, e apresentam resultados deveras interessantes. Existem algumas entidades de normalização internacional como o IEEE [2] (através do *Learning Technology Standards Committee* – LTSC), a ISO [3] ou iniciativas como a ADL - *Advanced Distributed Learning* [4] (através de modelos de referência como SCORM [5]) que formalizam conceitos e apresentam tecnologias relevantes na área da aprendizagem. Os conceitos *E-learning* [6] e B-learning [7] são exemplos da aplicação de sistemas de informação no âmbito da formação e aprendizagem, e tem vindo a apresentar inovações várias e vantagens interessantes em comparação com o sistema de ensino tradicional,

de entre muitas delas pode-se referir que torna possível a aquisição do conhecimento baseada em métodos e ritmos totalmente personalizados e a possibilidade de aprendizagem à distância [8].

Uma das aproximações possíveis dos sistemas de informação ao domínio da formação pode ter lugar numa das fases finais da aprendizagem: a fase da avaliação. Neste item, pode-se apontar três fases distintas que constituem o processo de avaliação: a preparação, a realização de provas e correcção. Consequentemente, é útil encontrar métodos para simplificar estas três etapas que estão associadas à fase de avaliação que, por vezes, é árdua e pouco eficaz na obtenção de uma avaliação justa e objectiva. Na realização de provas de conhecimento podem surgir várias modalidades: prova oral ou escrita, com ou sem consulta, realizada localmente ou à distância (com recursos a meios informáticos). Neste documento, são alvo de estudo apenas as provas escritas, realizadas num formato electrónico ou seja, provas de exame electrónicas. Uma vez neste domínio, pode-se segmentar estas provas em dois tipos distintos: o modo online e modo offline. A diferença principal destes dois tipos está no modo de realização da prova. Quando a prova é realizada à distância, significa que esta está localizada num servidor remoto e que, para a realizar, é necessário recorrer a uma estrutura de rede de forma contínua, durante toda a prova. Com este modo pode ainda ser possibilitado ao examinando o acesso e a consulta de informação na Internet. No caso da prova ser realizada localmente, durante a sua duração não existe necessidade da existência de uma estrutura de rede, sendo a prova submetida no final da sua realização. De modo geral, este último modo não proporciona o acesso ou a consulta de informação. Existem vantagens e desvantagens em cada uma destas modalidades, mas todas elas são utilizadas mediante a decisão do avaliador ou examinador, cuja intenção é adequar a prova à correcta avaliação de determinados conhecimentos dos indivíduos avaliados ou examinandos. Já existem algumas soluções no mercado que permitem não apenas a realização deste tipo de provas, mas também a disponibilização de cursos e conteúdos, o acompanhamento do progresso individual dos formandos, entre outras funções. Serve como exemplo o caso do software profissional da *Adobe Macromedia Breeze* [9], e os mais recentes sistemas de gestão da aprendizagem que organizam e disponibilizam cursos online através da *Web*, como é o caso da plataforma *CMS* (Course Management System) *Moodle* [10], ou *LMS* (Learning Management

System) [11], como o *GMAT* [12], *VUE* [13], *PROMETRIC* [14], entre outros. Estes três últimos sistemas são utilizados por algumas empresas que certificam formação, como por exemplo a *Cisco Systems* [15].

Na avaliação de qualquer prova electrónica, pode-se assumir a existência de factores externos perturbadores, que podem levar a uma avaliação incorrecta dos conhecimentos. Alguns motivos, como por exemplo a falta de preparação e a facilidade que pode existir na troca de informação entre alguns examinandos, levam a que estes tentem ultrapassar os limites definidos para um determinado exame, falseando as respostas e a sua classificação. Por outro lado, para os examinadores é difícil vigiar cada actividade nos computadores de exame e actuar no momento adequado. Estes dois factores representam um risco que ainda é potenciado caso a relação examinandos/examinadores seja elevada ou seja, para um exame para o qual é disponibilizado um número insuficiente de vigilantes, para o número de examinandos existentes.

Alguns dos sistemas apresentados possuem técnicas que permitem afastar estes factores como por exemplo os sistemas de teste acima enumerados. Contudo, são sistemas totalmente proprietários, dificilmente configuráveis a provas electrónicas com algumas especificidades e ambientes particulares, como por exemplo ambientes académicos fechados (provas electrónicas realizadas on-line mas apenas com a abrangência máxima de uma ou mais salas de exame). Assim, o propósito principal deste estudo foi o de encontrar uma solução adaptada para várias modalidades de prova, que ao mesmo tempo minimize os factores que interferem negativamente na avaliação e, consequentemente, minimize a necessidade de vigilantes para um determinado exame. Para controlar automaticamente estes factores, dos quais depende a eficácia dos exames e a correcta avaliação de conteúdos e dos examinandos, para reduzir a necessidade de vigilantes de um determinado exame e, consequentemente, facilitar a realização deste, a solução que este documento aponta, passa pela implementação de um método de monitorização electrónica, ou seja, realizada por computador. Através da monitorização electrónica, a interferência de factores externos, pode ser regulada de forma automática e aleatória, com capacidade adicional de seleccionar alvos críticos durante o exame, permitindo facilitar a vigilância dos exames por parte dos examinadores. Desta forma, pretende-se que esta solução contribua para o sucesso na realização de provas electrónicas, beneficiando todos os intervenientes do processo.

1.1 **- A Tese**

Num ambiente de formação (escolar, profissional ou académico), onde existe a necessidade de realizar a avaliação com recurso a uma prova electrónica, é por vezes difícil controlar um conjunto de factores indesejáveis, principalmente em provas onde existe um número considerável de examinandos. Nestes casos, assume-se que existe uma probabilidade elevada de existência destes factores que forjam os resultados da avaliação, e pode ser necessário mesmo incrementar proporcionalmente o número de vigilantes para que se consiga diminuir esta probabilidade.

Esta tese apresenta o resultado da investigação sobre esta problemática e como solução base propõe a utilização de monitorização electrónica. Através da utilização da monitorização por computador, estes factores indesejáveis poderão ser controlados e geridos de forma electrónica e automática. Pela especificidade das características exigidas ao sistema, e pela falta de adequação por parte dos sistemas existentes, esta investigação procurou um método de monitorização próprio para o tipo de prova referida, o que posteriormente levou à construção de um protótipo, e que permite vários modos de funcionamento, aplicando os conceitos abordados. Trata-se de um protótipo de monitorização automática, cuja função é processar informação relevante no âmbito da prova de exame, realizar a monitorização e actuar se for configurado para o efeito (por exemplo no caso da existência de actividades ilícitas pré-configuradas), e apresentar informação útil para o avaliador monitorizar e actuar na prova electrónica que se está a realizar. O protótipo pode portanto ter vários modos de actuação e, o seu plano de funcionamento pode ser definido no início da prova e redefinido durante a sua realização mediante parâmetros de configuração oferecidos ao avaliador. Mediante este plano de monitorização definido o avaliador pode adoptar uma postura mais passiva, deixando o protótipo actuar nos casos configurados para o efeito, ou então uma postura mais activa, configurando o protótipo para pré-monitorizar dados, sendo depois o avaliador um personagem activo e actuante nos elementos de risco. Assim, este sistema aponta uma solução concreta para as questões atrás enumeradas, proporcionando uma avaliação justa, eficaz na distribuição de vários recursos (humanos e materiais) e de execução simples.

1.2 **- Estrutura da Dissertação**

Em consonância com o que foi referido nos capítulos anteriores, este documento é dividido em duas partes fundamentais: a definição do método de monitorização, e o desenvolvimento de um protótipo.

No Capítulo 2 é apresentado o método de monitorização elaborado para a especificidade do contexto apresentado. Este capítulo inclui o entendimento do conceito de monitorização dinâmica, que serve de base para o desenvolvimento do protótipo de um sistema de monitorização.

No Capítulo 3 apresenta-se a especificação que esteve na origem do protótipo desenvolvido. Este capítulo inclui uma análise de requisitos e uma especificação funcional.

No Capítulo 4 é apresentado o conjunto de tecnologias que serviram de suporte ao desenvolvimento do protótipo e uma breve referência sobre a aplicação destas tecnologias no protótipo construído.

É no Capítulo 5 que se apresenta o processo de implementação do protótipo. Neste capítulo são descritas as etapas de desenvolvimento do protótipo e, ao mesmo tempo, são apresentadas as opções tomadas e aprofundada a aplicação das tecnologias associadas.

Depois disto, são apresentados no Capítulo 6, as conclusões e os resultados da implementação deste protótipo, onde se inclui as estruturas físicas necessárias ao seu funcionamento, algumas configurações e modelos de apresentação de informação.

No Capítulo 7 são apresentadas as referências deste documento, e no Capítulo 8 são apresentados os Anexos deste mesmo.

Capítulo 2 – A Monitorização Electrónica

O principal objectivo na utilização da monitorização é verificar a existência/inexistência ou regularidade/irregularidade de um evento num determinado ambiente. Existem muitas finalidades para a implementação de esquemas de monitorização e, quando estes envolvem pessoas, podem ser classificados como legítimos ou não legítimos. De forma geral, pode-se referir que a utilização da monitorização é legítima quando envolve a monitorização de um conjunto de acções num espaço próprio e com conhecimento prévio da pessoa que está a ser monitorizada. Por outro lado, a vigia sem fronteiras e sem conhecimento prévio dos intervenientes, pode levar à ilegitimidade da utilização da monitorização. Sendo assim, no caso da realização de uma qualquer prova electrónica que inclua um ou mais métodos de monitorização, é dever do examinador avisar todos os intervenientes de todos os processos de monitorização em curso.

Sob a orientação referida são apresentados, neste capítulo, os princípios orientadores para o método de monitorização e seguidamente o esquema de monitorização dinâmica adoptado.

2.1 **- Princípios Orientadores**

Como já foi referido neste documento, a necessidade de monitorização surge num contexto de formação, onde se pretende avaliar um conjunto de conteúdos através de uma prova electrónica. Quando é referido o termo prova electrónica, este implica que esta seja realizada com recurso a equipamento informático. O facto de ser uma avaliação por meio de uma prova electrónica, traz algumas vantagens na organização, tratamento dos dados e logística, como por exemplo:

- Facilidade na distribuição
- Possibilidade de realização remota
- Facilidade na recolha

- Facilidade na correcção

Ao mesmo tempo, existem alguns casos em que estes exames são realizados por um grande número de examinandos, e assim todo o processo de avaliação pode levantar questões relacionadas com atrasos e eficácia da prova. Por isso, convém assegurar que todos os examinandos sujeitos a avaliação cumpram um conjunto de regras definido, tornando a avaliação mais justa e eficaz, e ao mesmo tempo aplicar todo o processamento disponível para informatizar algumas tarefas e assim poupar tempo útil. Para ajudar neste propósito e de forma análoga à prova electrónica, recorre-se a uma monitorização electrónica de um conjunto de parâmetros no decorrer da prova, ou seja uma monitorização realizada através de um computador.

O método de monitorização que é implementado para conseguir estes propósitos foi pensado de forma a permitir a monitorização dois tipos de prova electrónica: as provas de realização local e as de realização à distância. De forma a detectar irregularidades na sua realização, foram estabelecidos critérios adequados para a monitorização, e definidas as variáveis centrais na recolha de informação. De várias variáveis de monitorização possíveis, optou-se por duas importantes e que se julga constituírem bons indicadores de factores externos e do normal funcionamento dos dois tipos de prova referidos, são elas:

- A Monitorização de Processos. Em qualquer computador com um determinado sistema operativo, a acção de execução de um determinado software, é traduzida num processo do sistema. Desta forma, a lista dos processos em execução [16], mostra em cada instante de tempo, o conjunto de recursos que o utilizador está a utilizar numa determinada tarefa. Se o utilizador está a resolver uma prova electrónica remota via uma ferramenta como um Navegador (*Browser*), então passa a existir na lista de processos do sistema, um processo chamado FIREFOX.EXE no caso do *browser* ser o *Firefox* da *Mozilla [17]*, ou o processo IEXPLORE.EXE, caso seja utilizado o *browser Internet Explorer* da *Microsoft [18]*. Qualquer outra acção que implique a remoção ou inserção de processos desta lista é portanto facilmente detectável. Um exemplo disto é a abertura de um ficheiro PDF (Portable Document Format) da *Adobe Systems Incorporated* [19], logo detectada pela consequente inserção na lista de processos, de um processo específico, o ACROBAT.EXE.

Este tipo de monitorização pode ser conseguido recorrendo a algumas funções presentes nalguns sistemas operativos, que permitem recolher informação e actuar sobre esta.

- A Monitorização de Ecrã Remoto. No âmbito de uma prova, muitas vezes é necessário visualizar as acções do utilizador. Com a monitorização do ecrã remoto, todo o ecrã visualizado pelo examinando é mostrado ao examinador. Este tipo de monitorização pode ser importante, em alguns contextos específicos de realização das provas referidas. Numa determinada prova, uma aplicação pode ser permitida dentro de determinados limite. Como exemplo, podem ser admitido o uso de browsers para consultar informação, mas pode não ser permitida a interacção em salas de conversação ou chats. Neste caso, a monitorização de ecrã permite que se observe comportamentos não autorizados que fazem uso de ferramentas permitidas. A monitorização de ecrãs apresentar-se-ia num computador central com acesso do examinador. Este tipo de monitorização pode ser conseguido através de ferramentas em regime de *Open-Source* [20], facilmente configuráveis para vários cenários e apresentações distintas. Uma das apresentações possíveis pode ser vista na Figura 2-1.

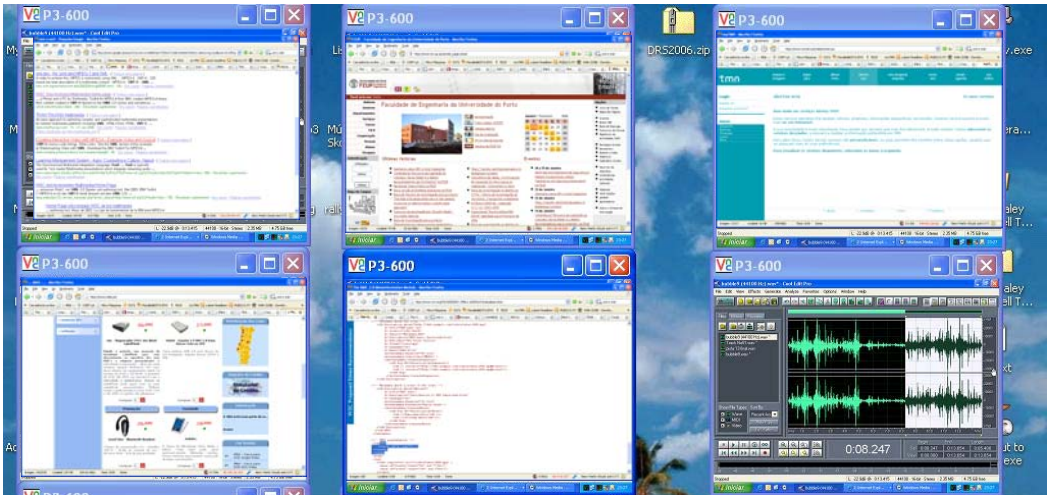


Figura 2-1 Exemplo de Apresentação – Monitorização de ecrãs

É sobre estas duas variáveis de monitorização que se baseia todo o sistema. A concretização prática destas soluções é apresentada mais à frente neste documento. Na secção seguinte é apresentado e descrito o método de monitorização dinâmica.

2.2 – O Método de Monitorização

A monitorização electrónica de uma determinada prova pode ser realizada de várias formas e assente em várias variáveis de monitorização. O método de monitorização que é apresentado neste documento é caracterizado por ser um método dinâmico, pois é pensado para alterar os seus critérios e parâmetros, tal como a forma de monitorização em qualquer altura, para conseguir adaptar o seu funcionamento, ao conjunto de factores que forem verificados durante o decorrer da prova. Sendo assim, pela dinâmica inerente ao sistema, terão de ser estabelecidos e actualizados vários programas ou planos de monitorização. O conjunto de programas estabelecidos durante a monitorização da prova vai ser doravante designado como o processo de monitorização.

No esquema apresentado na Figura 2-2, pode-se verificar que existem três recursos fundamentais que compõem o sistema com estas características base, são eles:

- O Examinador
- A Prova, Monitorização e Apresentação
- O(s) Examinando(s)

Figura 2-2 Esquema Geral de Monitorização – Recursos do Sistema

Note-se que, dependendo dos vários ambientes onde possa ser implementado este sistema, as entidades *Examinando* (objecto do exame, da avaliação) podem representar um indivíduo ou uma pluralidade de indivíduos sujeitos à avaliação, sobre os quais se desenrola a monitorização referida. Na forma mais simples e conceptual deste esquema podemos ter apenas uma entidade

Examinando. A partir deste ponto, neste documento serão encontradas referências a este esquema onde se faz esta simplificação.

O método de monitorização a ser implementado, reside inteiramente no bloco de *Prova, Monitorização e Apresentação* em evidência na Figura 2-2. Este bloco, por sua vez, é decomposto em três agentes distintos, como se mostra na Figura 2-3. Estes sub-blocos são chamados de agentes porque trabalham de forma autónoma com base na recepção e envio de mensagens. Na figura referida, pode-se verificar a forma como é estabelecida a comunicação entre estes três agentes, e a interacção com as duas entidades intervenientes no processo: o Examinador e o Examinando.

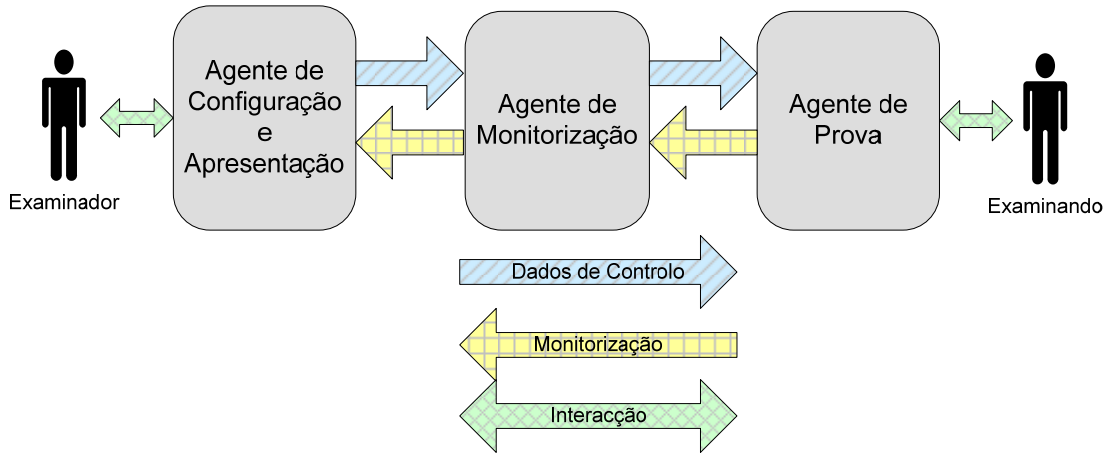


Figura 2-3 Método de Monitorização – Interacção de Agentes

O *Agente de Configuração e Apresentação* tem, como o próprio nome indica, a função de configuração dos planos de monitorização e a apresentação do resultado da monitorização. O *Agente de Monitorização* é aquele que realiza a monitorização do *Agente de Prova* seguindo os planos formulados pelo *Agente de Configuração e Apresentação*. A função do *Agente de Prova* é a de recolher e apresentar a informação pedida pelo *Agente de Monitorização*. Também na Figura 2-3, pode-se observar que entre os três agentes referidos, existem dois tipos de informação: os dados de controlo e a monitorização.

Os dados de controlo, como o próprio nome indica, carregam a informação para o controlo da monitorização. Estes podem ser dados de configuração ou dados de decisão. Os dados de configuração, podem ser passados ao *Agente de Monitorização* no início ou durante o decurso da

prova. No início é necessário que o *Agente de Configuração e Apresentação* contacte o *Agente de Monitorização* para configurar um esquema de monitorização e, depois de iniciada a prova, esta configuração pode ser alterada pelo Examinador caso este o pretenda fazer. Os dados de decisão são também dados passados do *Agente de Configuração e Apresentação* para o *Agente de Monitorização*, que transportam as decisões sobre determinado evento que não fora previamente configurado.

Repare-se que, estes dois dados de informação e controlo/decisão são trocados entre os Agentes periféricos (*Agente de Configuração e Apresentação* e *Agente de Prova*) e o *Agente de Monitorização.* Estes dados têm como objectivo realizar a monitorização propriamente dita, ou seja, transportar a informação das variáveis de monitorização e enviar eventos de decisão de acordo com o que estiver configurado.

Uma vez que se trata de uma monitorização dinâmica, por interacção humana (através da entidade examinador), ou de forma automática (pelo *Agente de Monitorização*), dá-se neste sistema a possibilidade do Examinador alterar a configuração deste, redefinindo os critérios da monitorização, alterando assim os factores de relevância para aquela prova. Neste sistema específico, existem duas formas de alteração do processo de monitorização: por interacção humana do examinador, tomando este as acções que considerar necessárias, ou por interacção automática do *Agente de Monitorização*, o que implica a execução de acções automáticas mediante dados de monitorização recebidos. Estas duas formas vão permitir três estados fundamentais, que podem ser observados no sistema, são eles:

- Autónomo – Estado onde apenas existem acções autónomas do *Agente de Monitorização* para o *Agente Prova*, conseguido após a configuração inicial do examinador. Este estado, apenas envolve o *Agente de Monitorização* e o *Agente Prova*, durante toda a duração da prova.
- Semi-autónomo – Estado de acções semi-autónomas, que envolvem três entidades, e que traduz um plano de monitorização com interacção parcial do examinador. Mesmo depois do início da prova, o examinador pode alterar os padrões de monitorização e actuar com base em dados que estão a ser monitorizados. Assim, o examinador, o *Agente de Monitorização* e o *Agente de Prova* interagem e alteram o plano de monitorização.

- Manual – Estado conseguido com uma configuração inicial, cujo processo decisório passa sempre pelo examinador e nunca pelo *Agente de Monitorização*. Neste estado nenhuma acção parte do bloco monitorizador, ou seja, este estado implementa uma simples monitorização de ecrãs e processos, só possível de ser alterada pelo examinador.

É baseado nestes estados de funcionamento e no tipo de estrutura apresentado, que este sistema dinâmico vai conseguir adaptar-se aos problemas colocados em diversas provas electrónicas. Este esquema permite uma gestão eficaz das informações que são veiculadas no sistema e ao mesmo tempo um gestão sobre as acções das duas entidades fundamentais: o Examinador e o Examinando. É sobre este esquema principal de monitorização que será construído o protótipo e por isso no próximo capítulo é dado o passo inicial no seu desenvolvimento: a especificação.

Capítulo 3 - Especificação do Protótipo de Monitorização

Neste capítulo é apresentada a especificação detalhada que surge depois de descrito todo o método de monitorização apresentado, referido no Capítulo 2. É no capítulo presente que é apresentado em primeiro lugar, a análise de requisitos do sistema. Estes requisitos são o resultado da integração do método de monitorização referido, com a criação das entidades que vão compor o sistema real. Em seguida, todas estas entidades são especificadas com os seus respectivos esquemas funcionais no tópico especificação funcional.

3.1 - Análise de Requisitos

Para que cumpra os objectivos da monitorização enunciados no Capítulo 2, o protótipo do sistema tem de ser desenvolvido sobre um conjunto definido e adequado de especificações. Este sistema, de acordo com os elementos referidos nos capítulos anteriores, deve incluir a existência de três agentes:

- O Agente de Configuração e Apresentação
- O Agente de Monitorização
- O Agente de Prova

Figura 3-1 Esquema Geral Básico – 3 Agentes

Tal como pode ser observado na Figura 3-1, cada um destes agentes que compõem o sistema, comunicam entre si para concretizar o método integral de monitorização. Estes agentes

representam, no fundo, uma máquina com processamento disponível, ou seja, um computador com características comuns, e são colocados numa estrutura de rede apresentada na Figura 3-2.

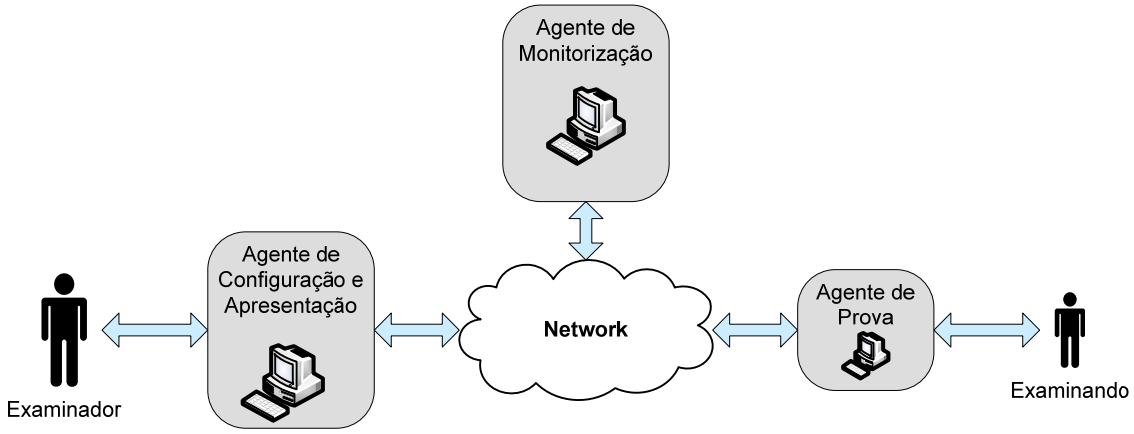


Figura 3-2 Esquema Geral Avançado e Estrutura de Rede – 3 Agentes

Note-se que as duas entidades, examinador e examinando são importantes para o funcionamento do sistema e por isso, mesmo que não sejam apresentadas, a sua presença é sempre necessária.

É através do *Agente de Configuração e Apresentação* que o examinador interage no sistema em três momentos: no início da prova, durante a realização da prova e no final da prova electrónica. Antes do início da prova, o *Agente de Configuração e Apresentação* deverá apresentar um conjunto de informações importantes para o sistema são elas:

- Informação de Autenticação: o acesso ao sistema será assim autenticado através de um nome de utilizador e uma palavra-chave de acesso. A partir daqui, é reconhecido ou não o direito a um examinador de gerir uma dada prova electrónica.
- Informação de Examinandos: assim é introduzido no sistema uma lista contendo os nomes dos examinandos que irão realizar a prova, a respectiva distribuição física na(s) sala(s) onde se realiza a prova, e o número identificador de máquina que lhe é designada (neste caso pode ser utilizado o número IP, número único de identificação numa rede de computadores). Estas informações deverão ser reconfiguráveis, permitindo ao examinador adicionar ou remover examinandos, acrescentar ou rectificar dados pessoais e alterar ou incluir a posição ou IP de um examinando no espaço onde se realiza a prova.

- Informação de Prova – desta forma pode ser inserido no sistema o modo de realização da prova. Esta informação deverá incluir vários elementos, nomeadamente a duração da prova, o tipo de processos importantes a monitorizar, os processos que serão permitidos ou não durante a sua realização ou se é uma prova com ou sem consulta.

Para o desenvolvimento deste sistema foram consideradas estas informações iniciais no entanto, outras informações poderão ser incluídas no sistema através da programação das interfaces e diálogos com a entidade examinador e a entidade examinando. O formato destes diálogos, a dinâmica de inserção destas informações e o tempo e a ordem em que estas são inseridas no sistema, não é objecto desta especificação, uma vez que não são determinantes para o sistema.

Depois de inseridas estas informações, e exclusivamente por ordem do examinador, são desbloqueados os Agentes de Prova e dá-se o início da prova e a sua monitorização. Nesta fase, deverão ser apresentados ao examinador dois tipos de informação, (de acordo com as variáveis de monitorização do sistema):

- Informação sobre os processos mais importantes que estão a ser executados nos computadores dos examinandos. Estes deverão ser apresentados em conjunto com algumas informações sobre o examinando, para que haja correspondência directa entre examinandos e processos.
- A visualização dos ecrãs de todos os computadores que estão a ser monitorizados, em regime de rotatividade, agrupados de diferentes modos escolhidos pelo examinador (em esquemas como 3x3, 4x3, etc).

Mesmo durante a duração da prova, deverá ser permitido ao examinador, a alteração instantânea de alguns parâmetros de configuração, como por exemplo no modo de apresentação dos dados no ecrã, na alteração da lista de processos permitidos/não permitidos, etc. Da mesma forma, deve ser também incluída a hipótese de visualização de dados mais específicos sobre determinado examinando. No ecrã apresentado ao examinador poderá também ser apresentado um espaço de destaque (espaço para um ecrã de maior proporção), que pode ser gerido automaticamente ou activado por ordem desta entidade. No modo automático, se o *Agente de*

Monitorização, verificar que existe alguma alteração à lista de processos que não está permitida, automaticamente coloca a janela desse computador em evidência, alterando a sua posição e proporção. No modo manual, é utilizador que comanda esta acção de colocar em evidência. Em qualquer dos casos, o ecrã deverá ficar em destaque, até que o examinador dê ordem para voltar à vista geral. Quando o ecrã está na posição de destaque, deverá surgir, ao mesmo tempo, uma indicação mais pormenorizada dos processos que o computador monitorizado está a correr, podendo o examinador tomar conhecimento destes e terminá-los se assim o considerar necessário.

Do lado do Examinando, mostrado na Figura 3-2, pode-se afirmar que o *Agente de Prova* tem como responsabilidade da apresentação dos dados ao *Agente de Monitorização* como já se tinha referido atrás. Este, logo de início, deve apresentar as condições de realização do exame definidas pelo Examinador e esperar as ordens para começar a prova. Depois disso, deve apresentar a prova ao Examinando, com a indicação do tempo que falta para o términos do tempo regulamentar. Findo este tempo, os Agentes de Prova podem ser automaticamente bloqueados de modo a fazer logo a entrega da prova ou esperar por ordens do examinador caso este queira avançar para um tempo suplementar.

3.2 **- Especificação Funcional**

Neste tópico, irá ser descrito o esquema funcional de todo o sistema e de todos os blocos que o compõem. Como já foi referido, a monitorização consiste em duas vertentes, a monitorização dos ecrãs remotos e a monitorização dos processos e admite a existência de três agentes fundamentais. Neste ponto do documento, torna-se necessário especificar com um maior detalhe como se deve comportar o sistema de forma global, para monitorizar e preconizar as acções referidas nos capítulos anteriores.

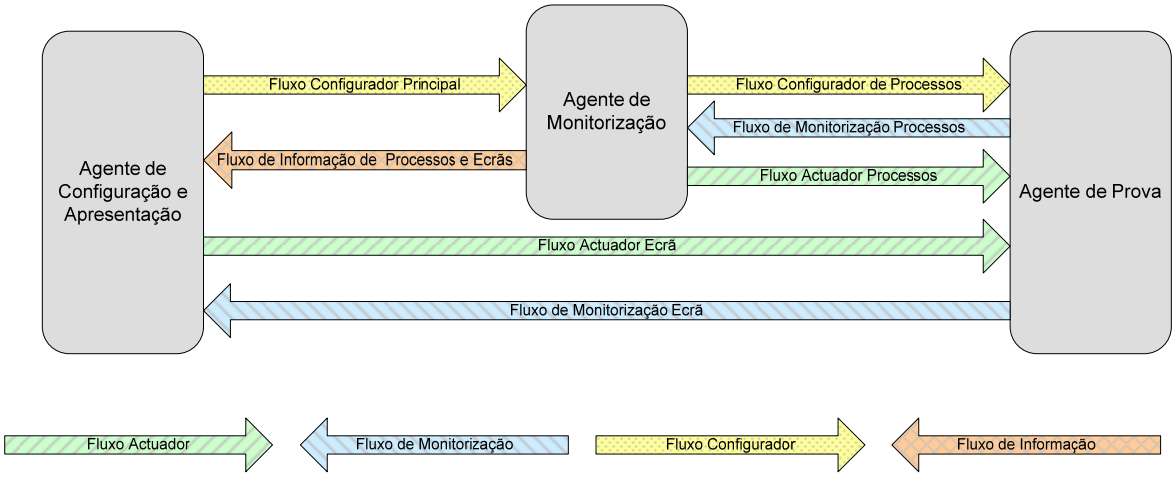


Figura 3-3 Esquema de Fluxos e Monitorização - A. de Monitorização – A. de Prova

Na Figura 3-3 é apresentado de forma global como é composto todo o sistema de monitorização. O sistema é composto por três elementos e vários fluxos de dados, designação que se utiliza neste documento para representar a informação que circula entre os vários blocos de forma esporádica ou regular no contexto da realização da prova electrónica. No sistema podem-se ver quatro tipos de fluxos: os fluxos actuadores, os fluxos de monitorização, fluxos configuradores e fluxos de informação.

Todo o sistema depende, logo de início, da existência de um fluxo de configuração principal enviado pelo *Agente de Configuração e Apresentação* que é dirigido ao *Agente de Monitorização*. Este fluxo contém as informações necessárias à realização da prova e ao tipo de monitorização a realizar. Este *Agente de Monitorização*, por sua vez, dá início à monitorização estabelecendo um fluxo de configurador de processos com o *Agente de Prova*. Este fluxo vai programar o tipo de eventos que, ao longo da prova, terão de ser emitidos pelo *Agente de Prova*.

A partir deste ponto a prova é iniciada e existe um fluxo de monitorização de processos que transporta o resultado dos eventos programados (no fundo, alterações à lista de processos em curso) no *Agente de Prova*. Esse fluxo é dirigido ao *Agente de Monitorização* que os interpreta e formula as acções de monitorização. Estas acções podem ser diferenciadas de duas formas: acções sobre processos e acções sobre ecrãs. As acções sobre processos que tiverem sido programadas antes do início da prova pelo examinador são enviadas directamente para o *Agente de Prova*, pelo fluxo actuador de processos para que automaticamente (sem a necessidade de

intervenção do *Agente de Configuração e Apresentação* e consequentemente do Examinador) possam ser terminados processos não admitidos na prova, por exemplo. Por outro lado, as acções sobre processos que não tiverem sido previamente programadas e as acções sobre ecrãs são enviadas em forma de eventos para o *Agente de Configuração e Apresentação* através do fluxo de informação de processos e ecrãs.

Mediante as informações recebidas através do fluxo de informação de processos e ecrãs, o *Agente de Configuração e Apresentação* estabelece cada fluxo actuador de ecrã directo ao *Agente de Prova*. Este origina um fluxo em sentido contrário surge o fluxo de monitorização de ecrã que é despoletado pelo fluxo actuador de ecrã e que segue para o *Agente de Configuração e Apresentação*.

Note-se que o Agente de Monitorização não tem qualquer influência no estabelecimento dos fluxos de monitorização de ecrãs que são apenas da responsabilidade do *Agente de Configuração e Apresentação*. A explicação para este facto advém deste fluxo ser um fluxo vídeo e, de forma a haver lugar à interpretação por parte do bloco monitorizador, teria de existir algum processamento de imagem laborioso que conseguisse identificar automaticamente irregularidades numa imagem do ecrã de um examinando e que assim permitisse em primeira instância a verificação electrónica e a actuação directa (nos seus processos, por exemplo). Esta é a razão para este processamento não ter sido contemplado no método de monitorização do *Agente de Monitorização* e assim, em última instância é o *Agente de Configuração e Apresentação*, que decide ou não iniciar o fluxo actuador de ecrã directamente para o *Agente de Prova*.

Da mesma forma, pela Figura 3-3 pode-se verificar que a informação acerca dos processos (lista de processos que não foram pré programados, informação de terminação ou execução de processos) e o fluxo de monitorização de ecrãs (eventos gerados) são apresentados sempre ao *Agente de Configuração e Apresentação* e, consequentemente, ao examinador da prova. Este, pode assim observar os eventos ocorridos e, se for o caso, reformular configuração do sistema ou enviar ordens pontuais através do fluxo configurador principal, ao mesmo tempo recebe o fluxo vídeo dos ecrãs dos *Agentes de Prova*.

Numa análise global inicial pode-se dizer que este é um sistema com as funções principais distribuídas e por isso, um sistema distribuído. Sendo o *Agente de Prova* o agente sobre monitorização e o *Agente de Configuração e Apresentação* o agente que consome essa monitorização, o Agente de Monitorização tem o papel importante de tornar automáticas algumas acções pré configuradas e assim facilitar todo o processo. Para que este sistema funcione, é necessária uma estrutura que permita a comunicação entre os três agentes e por isso também se torna importante que todos dividam funções e automatizem mecanismos para que não haja uma sobrecarga (alto volume de dados relativo) num dos troços da rede. Este facto é explicado com mais pormenor no Capítulo 5.

Os agentes apresentados na Figura 3-3 vão ser constituídos internamente por um ou mais blocos que implementam funções específicas e que entre si trocam informações para conseguir o esquema pretendido. A seguir é apresentada a especificação funcional de toda a dinâmica do sistema através da explicação detalhada dos blocos que constituem os três agentes apresentados e das comunicações que entre eles existem.

3.2.1 - Agente de Configuração e Apresentação

Para dar início à monitorização no sistema é necessário um plano inicial de monitorização. Este é construído no *Agente de Configuração e Apresentação* pelo Examinador e é conseguido através de duas informações básicas e importantes: a informação de descrição de ambiente e a informação de descrição dinâmica da cena, conforme se pode observar na Figura 3-4. Para construir esta informação, são disponibilizadas ao examinador duas interfaces: a Interface de Configuração de Ambiente e a Interface de Construção Dinâmica de Cena.

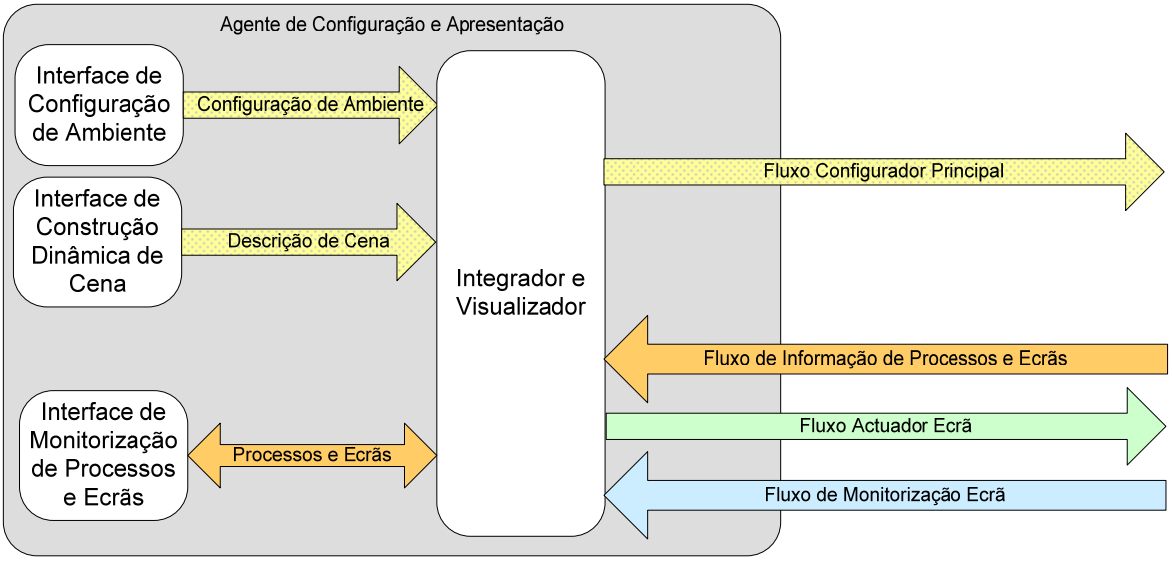


Figura 3-4 Agente de Configuração e Apresentação – Esquema Geral

A informação de configuração de ambiente deverá ser gerada previamente à realização da prova e deverá conter uma descrição de configuração do ambiente da realização desta, de acordo com o seguinte exemplo:

```
- ambiente a monitorizar: “Sala”
    - informação de identificação do aluno: “Pedro Pinto”
    - número identificador do computador (IP): “192.168.0.20”
    - posição relativa do computador no espaço: “15”
    - processos permitidos: explorer.exe; svchost.exe
    - processos npermitidos: messenger.exe; emule.exe
```

Código 3-1 Linguagem de Ambiente - Alto Nível

Quanto à informação de descrição de cena, esta deve apresentar-se como um fluxo constante que traduz no instante inicial os objectos que devem ser monitorizados e como eles se apresentam na cena de monitorização. Esta descrição é parte fundamental deste sistema e deve estar definida numa linguagem de descrição de cenas com suporte para cenas dinâmicas. Isto é, cenas onde os objectos visualizados mudam em função da decisão humana. Por um lado, a linguagem de descrição de cena terá de traduzir do modo mais fiel construções de cenas possíveis realizadas na “Interface de Construção de Cena”, por outro terá de ser uma linguagem adequada e consistente para a leitura no agente aos quais esta linguagem é dirigida, o *Agente de Monitorização*. Existem duas informações principais que são enviadas sob a linguagem de

descrição de cena. A primeira informação define o espaço de monitorização, com a definição da divisão do ecrã e a posição de cada janela de monitorização, exemplificado no código a seguir:

```
- Espaço de divisão do monitor: "4 Colunas 3 Linhas"
    - Janela número "1" com posição na "Linha 1 Coluna 3"
```

Código 3-2 Linguagem de Descrição Específica de Apresentação - Alto Nível

A segunda informação transporta dados sobre a dinâmica espacial e temporal da cena, apresentados como o seguinte exemplo:

```
- imagem do computador da posição "1" visualizado na janela "2" com início aos "4" segundos e
duração de "2" segundos
- imagem do computador da posição "15" visualizado na janela "4" com início aos "2" segundos e
duração de "4" segundos
```

Código 3-3 Linguagem de Descrição de Cena - Alto Nível

O conjunto destes elementos constitui um fluxo contínuo que traduz em cada instante a posição de todos os elementos na cena de monitorização. Uma vez que existem linguagens que se adaptam a esta descrição e ao problema que a sua aplicação coloca, existe a necessidade de uma análise comparativa que permita obter um resultado adequado à finalidade que se propõe. Por conseguinte, no tópico 4.3.2 é apresentado um estudo comparativo de um conjunto de linguagens possíveis de linguagens de descrição de cena para este contexto.

Depois de obtidos estes dois dados importantes para o sistema, estes são encaminhados para um bloco, chamado bloco Integrador e Visualizador. Este bloco tem a seu cargo várias funções, entre as quais, transferir a configuração inicial de Ambiente para a entidade seguinte que é o *Agente de Monitorização* e sincronizar a informação de Construção de Cena com este mesmo agente, sempre que esta informação seja inserida ou actualizada pelo *Agente de Configuração e Apresentação*.

Do lado do *Agente Monitorização* vem o fluxo de informação de processos e ecrãs. Por um lado, este fluxo transporta a informação sobre a acção nos processos que é dirigida ao bloco integrador e visualizador, onde esta informação sobre os processos é tratada e depois apresentada ao Examinador. Por outro lado é também ao bloco Integrador e Visualizador que se dirigem os fluxos que transportam a informação de ecrãs, e é com esta informação que este bloco estabelece o(s) fluxo(s) actuador(es) de ecrã com o(s) *Agente(s) de Prova*. Surgem então, como resposta a estes fluxos actuadores, os fluxos de monitorização de ecrãs respectivos, que

representam os fluxos vídeo de monitorização de cada ecrã, que são dirigidos ao bloco que os origina. Este agente pode estabelecer quantos fluxos de monitorização de ecrã pretender.

Finalmente é de salientar que este agente realiza funções de Configuração e funções de Apresentação como a própria designação indica e assim, considerou-se que desejavelmente este deverá ter estas duas funções separadas para que seja possível no futuro constituir um Agente de Configuração e um ou vários Agentes de Apresentação, para que seja mais fácil a monitorização de ecrãs realizada por vários examinadores, por exemplo.

3.2.2 - Agente de Prova

O *Agente de Prova* é o agente que está sempre associado ao Examinando e, ao mesmo tempo, é o agente de construção mais simples nesta fase. A Figura 3-5 apresenta o esquema de um *Agente de Prova* e daí pode-se verificar que este possui apenas dois blocos internos: o bloco de Monitorização de Processos e o bloco de Monitorização de ecrãs.

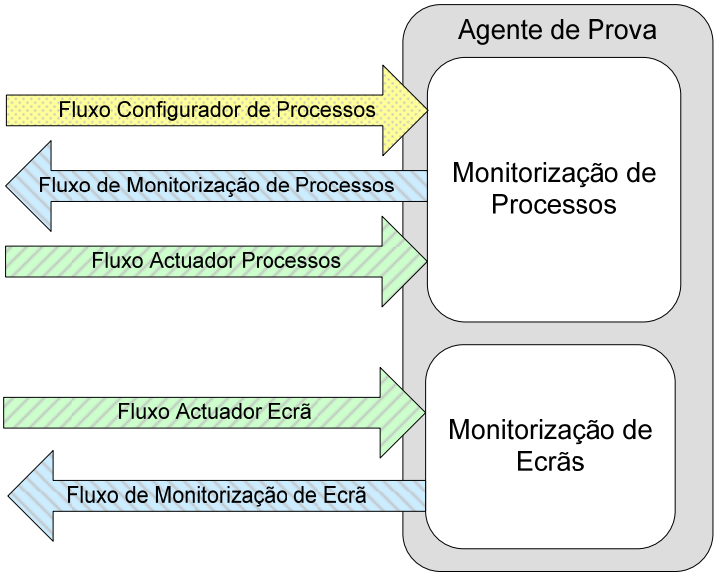


Figura 3-5 Agente de Prova – Esquema Geral

O bloco de Monitorização de Processos, como a própria designação indica, estabelece a interface para a monitorização dos processos que estão a ser executados no(s) *Agente(s) de*

Prova. É através desta interface que é possível um conjunto de funções que vão ser essenciais para o *Agente de Monitorização*, são elas:

- A comunicação da sua lista de processos em execução – quando lhe for solicitado (através do Fluxo Configurador de Processos), este bloco constrói a lista com todos os processos que estão a ser executados na entidade Examinando e/ou elabora relatórios com informações sobre quaisquer alterações à lista dos processos existentes e envia esta/estes para o *Agente de Monitorização* (através do Fluxo de Monitorização de Processos).
- A execução de ordens geradas no *Agente de Monitorização*, nomeadamente a execução ou terminação de determinados processos. Quando lhe for solicitado a execução ou terminação de algum processo, esta interface responde pela acção, executando esta mesma através do Fluxo Actuador de Processos e envia um relatório da execução ao *Agente de Monitorização* (através do Fluxo de Monitorização de Processos).

O outro bloco constituinte do *Agente de Prova* é o bloco de Monitorização de ecrãs. Esta recebe pedidos, através do Fluxo Actuador de Ecrã, para o estabelecimento de ligações que visam transportar o fluxo do vídeo referente ao *Agente de Prova* em questão, realizado através do Fluxo Monitorizador de Ecrã. Só a partir do momento em que é pedido a esta interface para estabelecer um fluxo de ecrã, é que esse mesmo ecrã do respectivo *Agente de Prova* passa a ser monitorizado visualmente. Durante todo o tempo da prova, os *Agentes de Prova* são monitorizados pelos processos que executam, mas apenas durante uma percentagem desse tempo são monitorizados através dos seus ecrãs. Esta percentagem depende de alguns critérios que poderão ser definidos no *Agente de Monitorização*. Como já foi referido, deverá ser possível configurar um regime de rotatividade em relação aos *Agentes de Prova* que estão a ser monitorizados. Além disso, também pode acontecer o facto de determinada acção de um *Agente de Prova* ser considerada irregular ou suspeita no decurso da prova. Este caso pode justificar o início de uma monitorização de ecrã a um determinado *Agente de Prova*, e assim realizada uma monitorização preventiva, colocando o ecrã respectivo em evidência (para a entidade examinador visualizar as suas acções suspeitas). Logo que qualquer um destes eventos tenha lugar (quer a monitorização rotativa, quer a monitorização preventiva), esta interface envia a informação visual que existe no seu ecrã para a entidade de origem desses mesmos eventos.

De seguida, vai ser descrito um Agente importante deste modelo - o *Agente de Monitorização* que interage com os dois Agentes até agora descritos.

3.2.3 - Agente de Monitorização

Na Figura 3-6 é apresentado o conjunto de blocos que compõem o *Agente de Monitorização*. Note-se que este agente é dividido em três partes: o bloco de Configuração e Apresentação, o bloco Monitorizador e o bloco Gestor de processos.

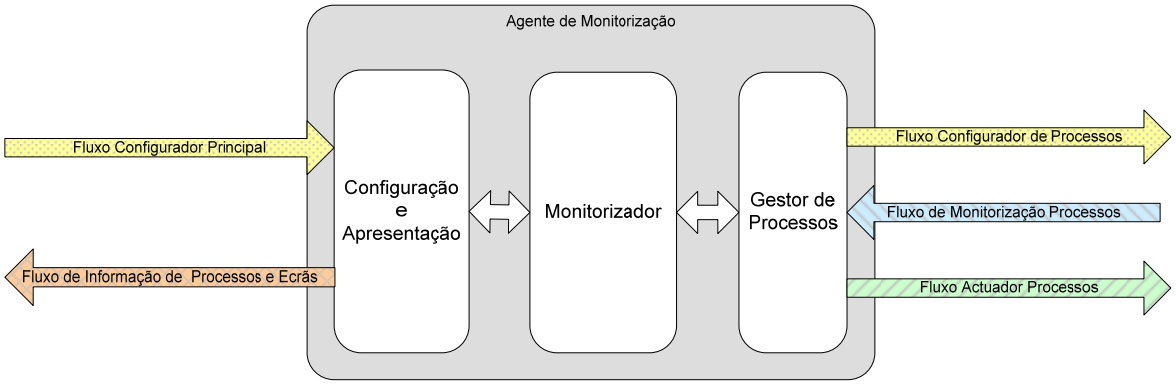


Figura 3-6 Agente de Monitorização – Esquema Geral

Todas as informações que partem do *Agente de Configuração e Apresentação* do bloco Integrador e visualizador (Configuração de Ambiente e Descrição de Cena) são dirigidas para o *Agente de Monitorização*, mais especificamente para o bloco de Configuração e Apresentação deste agente. Conforme pode ser visto na Figura 3-7, esta interface recebe os dados iniciais da configuração e os dados dinâmicos da descrição de cena, e passa estes dados para o bloco Monitorizador, no início da prova, como dados de configuração e, durante a prova, como dados de reconfiguração, caso haja essa indicação do *Agente de Configuração e Apresentação*.

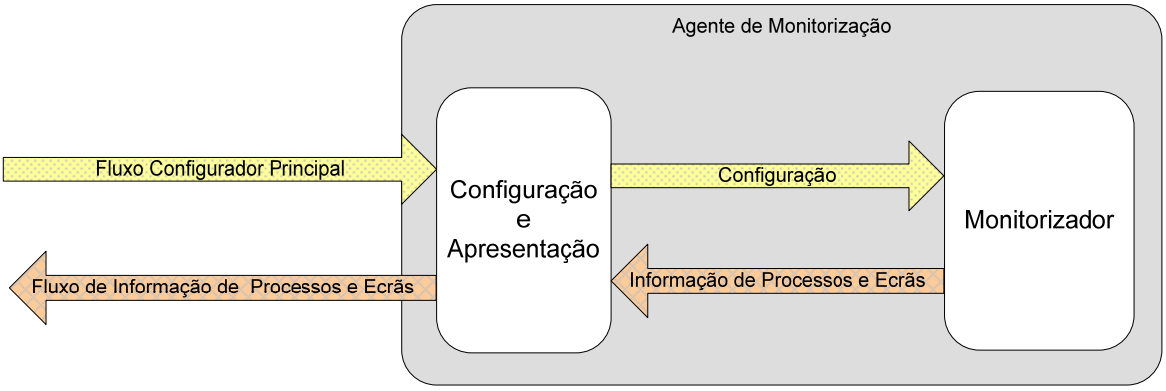


Figura 3-7 Interface de Configuração e Apresentação/Monitorizador – Dados de Configuração

O bloco Monitorizador é o importante gestor da monitorização dinâmica. Antes do início da prova ele guarda o plano de monitorização recebido e programa as suas funções para desempenhar as ordens que lhe estão confinadas. Mesmo ao longo da prova este bloco pode receber informações de reprogramação dos planos de monitorização. A sua função é cumprir os planos que lhe são enviados e comunicar ao *Agente de Configuração e Apresentação* a informação sobre processos e ecrãs (relativamente aos processos em particular esta informação segue sempre que não tenha acção predefinida para estes).

Aquando o início da prova, o bloco Monitorizador passa a configuração dos processos para o bloco Gestor de Processos, conforme é apresentado na Figura 3-8.

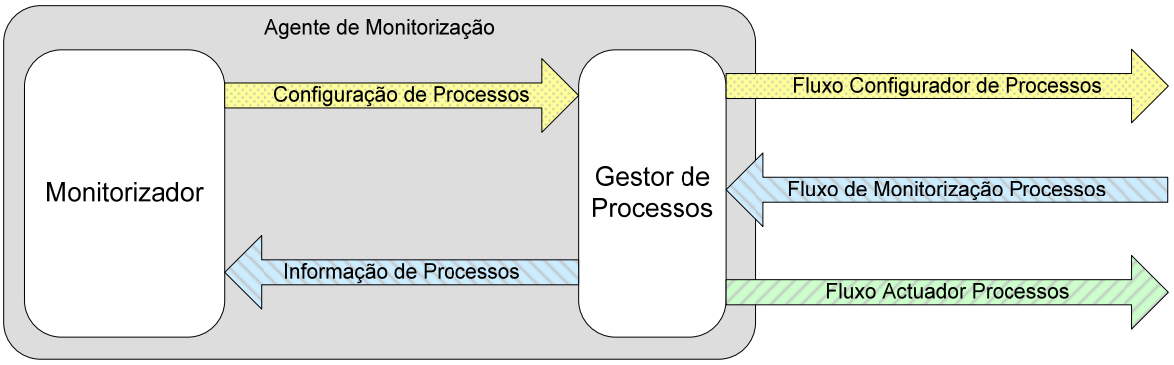


Figura 3-8 Agente de Monitorização – Gestão de Processos

Esta configuração é enviada como linguagem de ambiente cujo exemplo em alto nível pode ser visto no Código 3-1 e consiste na passagem de dois elementos básicos:

- A lista de processos permitidos – lista de processos que durante a prova podem ser executados.
- A lista de processos não permitidos – lista de processos que durante a prova nunca podem ser executados.

O bloco Gestor de Processos recebe a configuração de processos e vai realizar pedidos de listas ou pedidos de alarmes de acções relacionadas com os processos no *Agente de Prova*. Este conjunto de tarefas segue pelo fluxo Configurador de Processos. As respostas a estes pedidos, que podem ser listas de processos ou confirmação de acções, são enviadas de novo ao Gestor de Processos que procede à análise da informação recebida. Se tiver sido configurado para o efeito, então envia fluxos actuadores de processos para actuar directamente no *Agente de Prova*, no caso da terminação de um processo detectado que não é permitido. O bloco Gestor de Processos, tem a função de questionar os *Agentes de Prova* e fazer respeitar o conjunto de directrizes dadas pelo bloco Monitorizador. No decurso da prova, este bloco também deve passar informação importante ao bloco Monitorizador, no caso da ocorrência das seguintes situações:

- Processo não permitido executado (acção terminada) – Esta notificação serve para avisar o bloco Monitorizador que o *Agente Prova* executou um processo não permitido e este foi terminado. Poderá ainda existir a figura de processo não permitido executado (acção não terminada) – Esta notificação é semelhante à anterior, mas indica também que o processo não chegou a ser terminado (por razões alheias aos parâmetros de configuração).
- Processo Desconhecido. Neste caso, é avisado o bloco Monitorizador que o utilizador executou um processo que não estava presente em nenhuma das listas (lista de processos permitidos ou lista de processos não permitidos) e pede uma acção da parte do Monitorizador.

Em qualquer altura, o bloco Monitorizador poderá actualizar estas duas listas enviadas inicialmente, ou seja, alterar a lista dos processos permitidos e não permitidos.

Nos dois casos referidos em que o bloco Gestor de Processos notifica o bloco Monitorizador, este executa os seguintes procedimentos:

- Actua automaticamente sobre determinado evento que tenha sido notificado, caso tenha sido programado para o efeito, através da configuração que é passada pelo *Agente de Configuração e Apresentação*.
- Envia a informação da notificação para o bloco de configuração e apresentação, de forma a ser apresentada ao *Agente de Configuração e Apresentação*.

O *Agente de Monitorização* tem uma função importante além do que já foi referido: com base nas notificações que recebe do Gestor de Processos, o *Agente de Monitorização* oferece ao *Agente de Configuração e apresentação* um fluxo contínuo de informação para monitorização de ecrãs. Esta informação é passada pela informação de processos e ecrãs, conforme se pode ver na Figura 3-9.

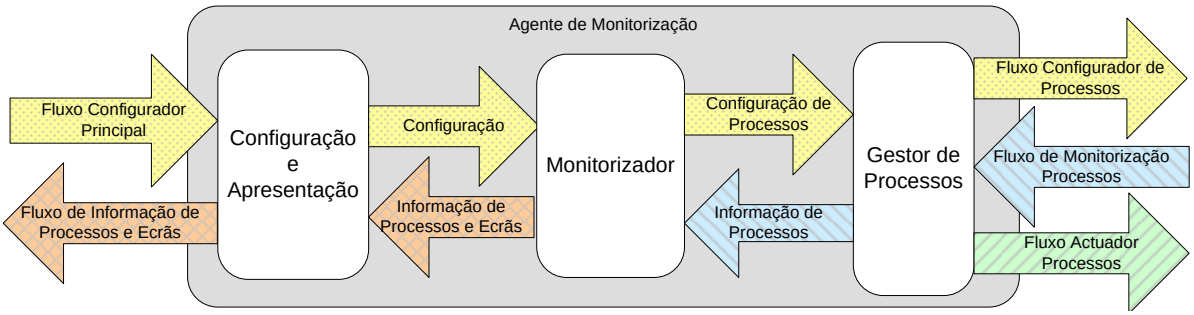


Figura 3-9 Diagrama de Blocos – Interface, Monitorizador e Gestor de Processos

A informação de processos e ecrãs que é passada pelo bloco Monitorizador para o bloco de Configuração e Apresentação, e consequentemente para o *Agente de Configuração e Apresentação*, transporta um fluxo contínuo de acções para o estabelecimento dos fluxos de vídeo entre o *Agente de Configuração e Apresentação* e o *Agente de Prova*, com o tipo de rotatividade configurada. Um conjunto possível é apresentado no Código 3-4.

```
Inicia      Ecrã 10    Posição 1:3
Termina     Ecrã 09    Posição 1:4
Processo    NPermitido: messenger.exe
Inicia      Ecrã 02    Posição 1:4
Inicia      Ecrã 12    Posição 1:2
Termina     Ecrã 10    Posição 1:3
Processo    Desconhecido: plus.exe
Termina     Ecrã 04    Posição 1:7
```

Código 3-4 Informação de estabelecimento de fluxos vídeo

Por último, convém reforçar que o bloco de Configuração e Apresentação tem com função, não só receber o fluxo configurador principal e enviá-lo para o bloco Monitorizador, mas também

receber estes dados da monitorização dos ecrãs do bloco monitorizador e enviá-los ao *Agente de Configuração e Apresentação*.

A partir do momento em que se dá ordem para o início da prova, todos os três blocos no *Agente de Monitorização* trabalham em conjunto para implementar a dinâmica de monitorização configurada. No Capítulo 8 é apresentado um esquema global desta dinâmica entre os três Agentes agora especificados.

No próximo capítulo, são apresentadas as tecnologias de suporte que permitiram a implementação do protótipo de monitorização assente nos conceitos descritos no capítulo presente.

Capítulo 4 - Tecnologias de Suporte

No seguimento do capítulo anterior, depois de referido o conjunto de especificações que o protótipo deve incluir, apresenta-se neste capítulo um conjunto de tecnologias que, na fase da especificação e implementação, serviram para implementar o protótipo. Estas tecnologias serão explicadas e referenciadas para melhor compreensão de todo o documento.

4.1 - Linguagens de Programação – Perl e PHP

Uma das tecnologias utilizadas para o desenvolvimento do protótipo foi a linguagem de programação Perl [21]. Na construção dos componentes do sistema de monitorização, surgiu a necessidade de recorrer a uma linguagem de programação. A escolha desta linguagem de programação foi baseada num conjunto de requisitos e vantagens que a linguagem Perl satisfaz e submete, nomeadamente o facto de ser uma linguagem:

- Sem necessidade de compilação, tornando o processo mais simples e independente de compiladores
- Simples e versátil para o trabalho em causa (tratamento de informação em formatos como XML ou SMIL facilmente reconhecíveis)
- Integrada e modular, com possibilidade de utilização e construção de módulos para o sistema a implementar
- Caracteristicamente inter-sistemas ou seja, permite que o código seja implementado de forma transparente em diferentes sistemas operativos

A vantagem do Perl relacionada com a possibilidade de integração de alguns módulos importantes, demonstrou ser muito útil na fase de testes. Nesta fase, alguns módulos foram escolhidos para adequar melhor esta ferramenta ao sistema. Os módulos mais importantes para o contexto actual deste documento são: o módulo *XML::Simple*, o módulo *Getopt::Long* e o módulo *Win32::API*.

Depois da fase de testes, deu-se início à fase de instalação definitiva deste conjunto de módulos no sistema (no tópico 8.2 pode ser visto um exemplo de instalação de módulos - a instalação do módulo *Win32::API*).

A linguagem Perl demonstrou ser, logo de início uma linguagem adequada ao sistema e a sua utilização permitiu construir vários blocos, apresentando resultados interessantes que vão ser apresentados posteriormente neste documento.

Outra das ferramentas essenciais para o projecto foi o PHP. O PHP (*PHP: Hypertext Preprocessor*) [22] é uma linguagem que permite a produção de páginas *Web* dinâmicas. As mais recentes versões desta linguagem conferem-lhe uma robustez e uma larga capacidade de integração com vários tipos de tecnologias como por exemplo, bases de dados (com módulos específicos *MySQL*, e outros), ferramentas multimédia, etc). Uma vez que a linguagem Perl pertence à origem do PHP, pode ser construídos sistemas com interfaces Web dinâmicas em PHP que produzam ou consumam informações de e para um sistema com linguagem Perl. Além destas características importantes, o PHP permite a integração do protocolo *SOAP* [23], permitindo a troca de mensagens em XML numa rede de computadores. No desenvolvimento do protótipo pretendido foi utilizada uma versão integrada de Servidor Web, Base de dados e PHP chamada XAMPP [24].

4.2 **- Monitorização do Ecrã (VNC)**

Tal como a especificação o indica, o protótipo inclui a monitorização dos ecrãs remotos. Este procedimento implica o recurso a ferramentas de software que permitam a visualização de ecrãs remotos. Nesta área existem algumas soluções que implementam esta dinâmica como é o caso do VNC (*RealVNC* [25]), o *PCAnywhere* (*Symantec* [26]) ou o *GoToMyPC* (*Citrix* [27]). Um dos propósitos mais relevantes para a decisão de uma solução para o protótipo é a interoperabilidade entre sistemas e o regime não proprietário do código fonte. Para o âmbito pretendido, o carácter não proprietário apresenta-se como um critério preponderante na escolha de uma destas soluções, uma vez que existirá a necessidade de aceder/alterar as interfaces deste software só disponíveis através do acesso/alteração do seu código fonte. Sendo assim, optou-se por uma

solução: o software VNC. O VNC (*Virtual Network Computing*) é um software da autoria da equipa da RealVNC que apostou em desenvolver, melhorar e promover conceitos de comunicação remota, dando origem a várias versões do software referido com características interessantes no âmbito deste trabalho, nomeadamente:

- O facto de constituir um pacote pequeno, simples de instalar e utilizar
- O facto de possibilitar a visualização e interacção com computadores remotos através de plataformas (Sistemas Operativos) independentes.
- Ser livre segundo licença GNU [28]. Este software está disponível em três versões: *Free Edition*, *Personal Edition* e *Enterprise Edition*. Destas três versões, apenas a primeira é distribuída pela licença GNU, sendo as restantes, versões comerciais. No âmbito deste trabalho foram analisadas e testadas duas versões deste servidor: a versão *Free Edition* e a versão *Enterprise Edition* (a análise desta última versão foi conseguida porque, apesar de ser uma versão comercial, é distribuída em versão de experimentação em 15 dias).

As potencialidades deste software são várias, entre as quais pode-se salientar: a possibilidade de tomar o controlo completo de máquinas remotas, fazer assistência técnica activa num computador remoto ou a fazer a monitorização de um grupo de pessoas (estudantes por exemplo), permitindo também realizar demonstrações colectivas de procedimentos, apoio educacional ou monitorização de acções.

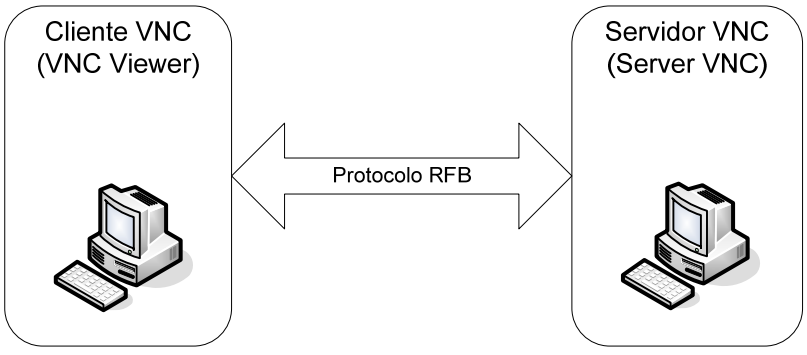


Figura 4-1 Diagrama do VNC – Arquitectura Cliente-Servidor

Como se pode observar na Figura 4-1, qualquer destes modos de funcionamento são conseguidos através de uma arquitectura cliente-servidor, onde um código específico é instalado

em ambas as partes, não necessitando estas de possuir o mesmo sistema operativo (pode ser utilizada uma máquina em Windows para monitorizar uma máquina em Linux, ou vice-versa). A comunicação é realizada por um protocolo específico, explicado mais à frente neste documento.

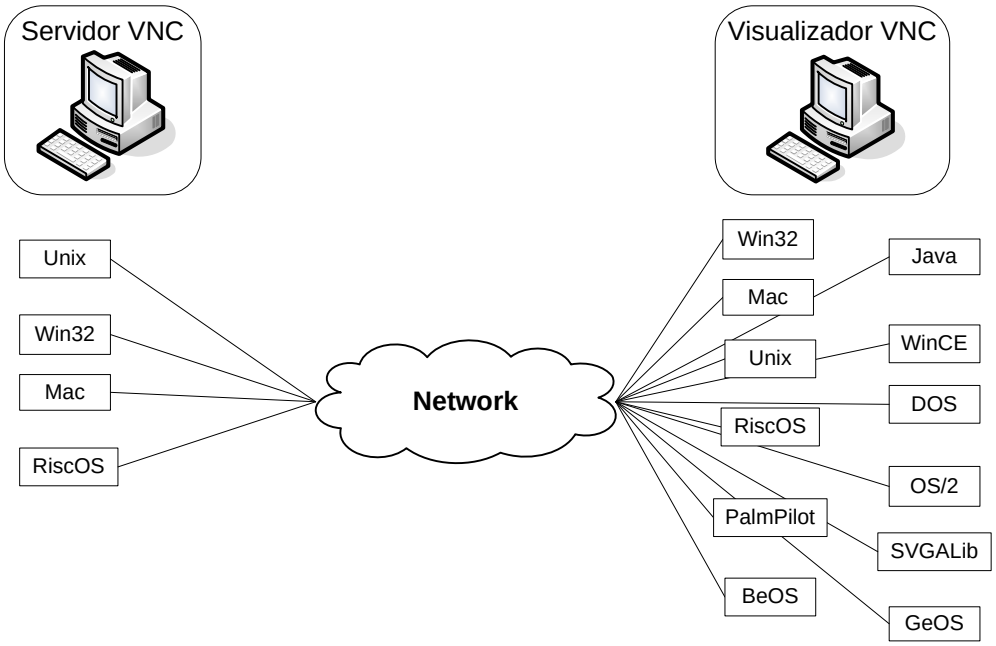


Figura 4-2 VNC – Interoperabilidade de Sistemas (Baseada em imagem de RealVNC [25])

Na Figura 4-2 podem ser vistas várias alternativas de comunicação entre vários sistemas, utilizando a arquitectura referida, através de uma rede de computadores. A comunicação é iniciada pelo cliente e dirigida ao servidor. A função do computador remoto (servidor) é enviar o fluxo actualizado do seu ecrã para o cliente. O cliente, por sua vez, apresenta este fluxo no seu próprio ecrã. Este processo pode ainda envolver interacção do cliente no computador servidor, ou seja, o software permite a envio de eventos do teclado ou acções do rato para a máquina remota.

Todo o pacote de instalação é bastante pequeno e fácil de instalar e configurar. Além disso, exige requisitos mínimos que facilmente são cumpridos pela generalidade dos computadores que possuímos nos dias de hoje, mesmo quanto à compatibilidade entre os vários tipos de hardware existente (placas gráficas, teclados, ratos, etc). O seu desempenho para o caso presente é bastante positivo, e, na fase de testes do protótipo, esta escolha, mostrou ser bastante fiável mesmo nas piores condições de conectividade, principalmente devido à baixa transferência de

informação que propõe com o protocolo RFB [29] (protocolo específico implementado pelo servidor VNC), e devido ao protocolo de actualização adaptativa (*Adaptive Update Protocol*) que possuí. Este aspecto é muito importante pois, numa situação real, serão estabelecidos tantos fluxos VNC, quantos os examinandos que existirem para uma determinada prova. Por um lado, é desejável que o tráfego que se acaba por gerar não sobrecarregue a rede e, por outro lado, é desejável que todos os fluxos que forem criados se adaptem facilmente às condições de rede que existirem.

Uma vez que o código fonte da versão *Free Edition* está disponível sobre a linguagem java, é possível submeter alterações ao seu modo de funcionamento para que esta ferramenta se adapte a várias formas de monitorização que podem ser assumidas.

Conforme já referido, os dois componentes de um sistema VNC são o visualizador e o servidor. Nos tópicos seguintes, estes dois componentes são descritos com maior profundidade.

4.2.1 - O Visualizador VNC

O visualizador VNC (ou VNC viewer) é um dos componentes do software VNC, que se instala num qualquer computador e que permite elaborar um pedido para um fluxo de ecrã remoto. Esse pedido pode ser realizado de várias formas, são elas:

- Através do executável VNC – Neste caso será necessário executar o ficheiro vncviewer.exe (Windows) ou vncviewer (Linux). Antes da ligação ser efectuada, este componente pede algumas informações, como por exemplo o endereço do componente servidor, ou a proporção do ecrã a apresentar. Depois disso, no componente servidor é formado o fluxo de vídeo dirigido ao visualizador. Este modo pode ainda ter uma variação: este ficheiro pode ser executado a partir da consola e pode ser executado com parâmetros na chamada do ficheiro, indicando as informações necessárias/pretendidas.
- Através de um ficheiro de configuração VNC - desta forma, pode ser gerado um ficheiro com as informações necessárias à ligação com a componente servidor, ficheiro este que tem a extensão “vnc” e poderá, em qualquer altura, ser executado. No Código 4-1 pode

ser visto um exemplo de um ficheiro de extensão VNC, com todos os parâmetros para iniciar um fluxo VNC do servidor para o visualizador.

- Através de um navegador *(browser)* com *Java* – Neste modo, o cliente ou o computador visualizador não necessitam de possuir qualquer código para conseguir a ligação ao servidor. Basta para isso utilizar um browser, colocar o endereço do servidor, e descarregar um "java applet" que o servidor envia. Depois disso aparecem as várias configurações possíveis e a hipótese de dar início ao fluxo de ecrã.

```
 [Connection]
Host=192.168.0.15
Password=3e4d64286ccfee00
Encryption=Server
[Options]
UseLocalCursor=1
UseDesktopResize=1
FullScreen=0
FullColour=1
LowColourLevel=1
PreferredEncoding=ZRLE
AutoSelect=1
Shared=0
SendPtrEvents=1
SendKeyEvents=1
SendCutText=1
AcceptCutText=1
DisableWinKeys=1
Emulate3=0
PointerEventInterval=0
Monitor=\\.\DISPLAY1
Scaling=30%,None
MenuKey=F8
AutoReconnect=1
```

Código 4-1 Exemplo de um Ficheiro de Configuração do VNC

4.2.2 - O Servidor VNC

O servidor VNC é um componente fundamental em todo o processo. A monitorização dos ecrãs referida no âmbito deste trabalho, é conseguida com recurso ao protocolo RFB que trata de estabelecer fluxos deste protocolo para cada computador a monitorizar, fluxos estes que transportam todos os *pixels* do ambiente de trabalho visto por cada utilizador desses mesmos computadores.

Aquando a concepção, os componentes VNC (o visualizador VNC e o servidor VNC) foram desenvolvidos para que o cliente fosse o mais simples possível, acarretando para o servidor um grau de complexidade mais elevado para cumprir os objectivos da plataforma. Ambas as versões experimentadas (versão *Free Edition* e *Enterprise Edition*) revelaram ter potencialidades bastante interessantes. Sabendo disto, e ao analisar a fundo o servidor VNC, pode-se verificar um

grande número de opções e configurações possíveis. Como exemplo, pode-se referir as várias opções de configuração que permitem manipular ambiente, a manipulação de janelas (tal como fazer o "Docking" automático), as configurações de segurança e encriptação (com passwords e protocolos de encriptação), entre outras. Em baixo na Figura 4-3, é apresentado um esquema onde se pode ver a interacção entre o Visualizador e o Servidor VNC.

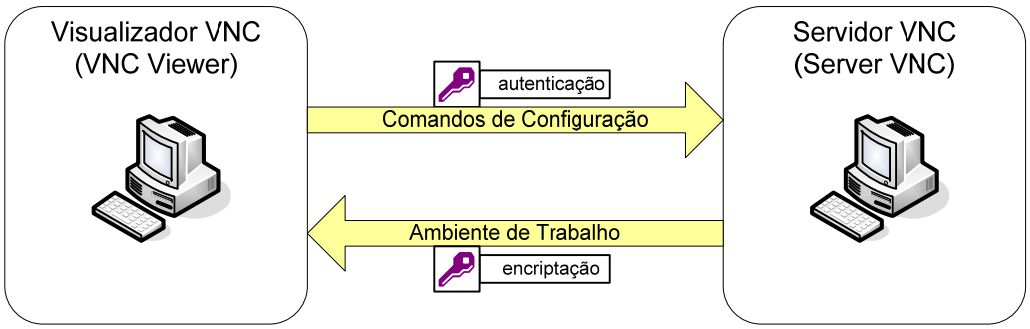


Figura 4-3 Esquema de Autenticação e Encriptação - Cliente-Servidor

4.3 - Linguagens de Descrição (XML)

Como linguagem de descrição de alguns elementos essenciais para a construção do protótipo, foi utilizada a linguagem XML [30], linguagem normalizada pelo W3C [31]. Uma parte considerável no desenvolvimento deste protótipo assenta em linguagens de descrição de cenas e ambientes e por isso, a escolha desta linguagem foi determinante. O XML possui características adequadas no contexto apresentado nomeadamente:

- É uma linguagem de norma não proprietária
- É independente dos vários tipos de plataformas e sistemas, permitindo um alto grau de interoperabilidade;
- Oferece capacidade descritiva adequada para ferramentas de geração de descrições automáticas ou para uma edição simples em modo texto com linguagem perceptível para o ser humano;
- Permite uma descrição estruturada de meta-informação para selecção, avaliação, e integração de objectos;

- É apropriada para hierarquias de elementos que representam vários tipos de objectos e conceitos adequados a sistemas gráficos e cenas;

É criada então nesse contexto, uma interface de geração de cena que tem como função apresentar uma *GUI* (Graphical User Interface) ao utilizador do sistema, para que se possam configurar ou verificar os parâmetros de uma determinada monitorização. É esta entidade que vai gerar a descrição de cena em XML consoante as opções inseridas.

Apesar desta dinâmica ser essencial ao protótipo, também no mesmo formato vão ser apresentadas várias outras descrições com informação para o Configurador, como por exemplo, informação referente às Configurações Básicas de Monitorização.

4.3.1 - Linguagens de Descrição de Ambiente

Conforme já foi referido, um dos desafios apresentados neste trabalho foi o de encontrar um formato adequado na informação das descrições, neste caso, a descrição de Ambiente. Depois de decidida a linguagem XML para o protótipo, no seu âmbito terá de ser definida uma sub linguagem para representar da melhor forma, o conjunto de elementos no ambiente a monitorizar. Trata-se de uma informação estática mas que exige uma representação:

- Não proprietária
- Intuitiva para edição e alteração
- Com capacidade de interoperabilidade entre plataformas e sistemas

Da análise efectuada para este tipo de descrição, as inúmeras vantagens do XML - *X3D* demonstraram ser importantes neste contexto, e esta linguagem foi adoptada uma vez que as suas características encaixam nas necessidades que se colocaram na fase da especificação.

4.3.2 - Linguagem de Descrição de Cena

Neste tópico vai ser descrito o processo de adopção de uma linguagem de descrição de cena, adequada para o sistema. Das várias linguagens baseadas em XML que podem ser encontradas para esta componente essencial da aplicação referida, apenas vão ser referenciadas neste documento aquelas que, por um lado, possuem um uso mais corrente e são aplicadas num

número considerável de aplicações com um grau de desenvolvimento e maturidade bem definido e, por outro lado, que se enquadram sob os aspectos gerais na aplicação de monitorização desenvolvida. Cada uma das linguagens analisadas incorpora um conjunto de mecanismos próprios, que resultam na obtenção de diferentes resultados. Nesta fase torna-se imperativo descrever de uma forma minimalista algumas linguagens de descrição de cena que, pelo estudo dos pré-requisitos efectuado, se apresentaram como as soluções mais adequadas ao sistema, realçando as suas características individuais relevantes ao contexto. Segundo este propósito, seleccionaram-se as seguintes linguagens:

- XMT-A [32]
- XMT-O [33]
- SMIL [34]

Segue-se uma descrição breve destas linguagens e suas características relevantes. Nos tópicos seguintes a designação SMIL depreende a referência à versão 2.0, ou seja, SMIL 2.0 [35].

4.3.2.1 - Linguagens desenvolvidas no contexto da norma MPEG

A norma *Moving Picture Experts Group* [36] define uma estrutura onde é possível transmitir vários tipos de objectos numa plataforma comum. Esta norma divide-se em duas partes fundamentais: a codificação dos objectos intervenientes e a possibilidade de definição da apresentação dos objectos pertencentes à cena multimédia. A apresentação, dos objectos que compõem a cena audiovisual, é especificada através de formatos próprios para descrição das cenas. O fluxo que passa este tipo de interacções e que contém as informações das relações entre os objectos multimédia, é chamado fluxo descritor de cenas e é especificado através da linguagem *BInary Format for Scene* (BIFS) [37]. O BIFS foi inspirado na linguagem VRML [38], e possui características próprias, adequadas para o sistema de transporte MPEG-4 [39]. A linguagem BIFS permite a eficiente representação de apresentações dinâmicas e interactivas, incluindo gráficos 2D e 3D, imagens, texto e material audiovisual. Esta representação inclui a descrição de organização no espaço e no tempo dos diferentes elementos de cena, animações e interacções com o utilizador. Ao contrário do VRML, onde os seus objectos e eventos são declarados em modo texto, em BIFS esta estrutura é especificada em código binário. A adopção

do formato binário é fundamental para a compressão e posterior distribuição das cenas no MPEG-4 no entanto, esse formato dificulta o processo de criação e alteração do conteúdo, embora este facto possa ser contornado com a utilização de ferramentas de edição gráfica, onde o formato binário já se torna transparente para os editores destes conteúdos.

Com intuito de ultrapassar a limitação do formato binário na criação e alteração de conteúdos na linguagem BIFS, o formato eXtensible MPEG-4 Textual (XMT) [40] foi proposto como uma estrutura para especificar descrições de cenas MPEG-4 através de uma sintaxe textual. O XMT é uma extensão textual usada na norma MPEG-4 e tenta combinar vantagens da representação binária BIFS e da representação de alto nível de abstracção de cenas multimédia, SMIL. Sendo assim, o XMT apresenta características novas no contexto MPEG-4 como:

- Um intercâmbio transparente entre editores e ferramentas de produção
- Interoperabilidade entre formatos. O formato XMT pode ser convertido em SMIL, VRML ou em BIFS (para os leitores MPEG-4)
- Possibilidade de edição manual do conteúdo

Esta estrutura é instanciada por duas linguagens, com diferentes sintaxes e semânticas XMT-A e XMT-O apresentadas de seguida. De seguida é apresentado um exemplo de uma descrição de cena no formato XMT-A.

```
<Shape>
   <geometry>
      <Rectangle size="40 30"/>
   </geometry>
   <appearance>
      <texture>
         <MovieTexture url="http://site.pt/obj.mpg" loop="false" startTime="0.0" stopTime="-1.0"/>
      </texture>
   </appearance>
</Shape>
```

Código 4-2 Código Exemplo da linguagem XMT-A

A linguagem XMT-A permite a descrição de baixo nível de conteúdos multimédia baseada em Extensible 3D (X3D) [41] e é uma versão em formato texto da linguagem BIFS. A sua estrutura baseia-se em construções XML com representação directa das expressões existentes no formato binário do MPEG-4, e por representar construções existente no BIFS, o XMT-A também se aproxima do tipo de estruturas definidas em VRML. Também no aspecto semântico, esta linguagem apresenta uma equivalência estrita com o BIFS, não acrescentando nenhuma outra

funcionalidade adicional. Assim sendo, apresenta o mesmo conjunto de características centrais já oferecidas pelo formato binário, que se apresentam enumeradas desta forma:

- Permite a integração e controlo de conteúdos áudio/vídeo diferentes na mesma cena.
- Tem um conjunto rico de construtores gráficos 2D/3D.
- Possui um conjunto de elementos que permitem a interactividade local e remota.
- Permite animações locais e remotas como por exemplo a alteração de posições de objectos, de cores, e formas.
- Permite a reutilização de conteúdos através de referências para fluxos.

Como se pode observar, existe um alto grau de verbosidade para o tipo de exemplo simples apresentado no Código 4-2.

Com o intuito de diminuir a complexidade existente na linguagem XMT-A na especificação de cenas MPEG-4, foi desenvolvida a linguagem XMT-O. Nesta linguagem, os objectos audiovisuais e as suas transformações são descritas num alto nível de abstracção, facilitando o processo de edição, ao contrário do que ocorre em XMT-A. A linguagem XMT-O tende a uma aproximação forte à linguagem SMIL e por isso apresenta algumas das suas características próprias, conseguindo assim uma interoperabilidade entre estes dois sistemas. Isto é conseguido pelo XMT-O recorrendo a alguns módulos presentes na linguagem SMIL. A estrutura do documento XMT-O também é dividida em duas partes, um cabeçalho e um corpo. No cabeçalho encontram-se as informações gerais do documento, como a forma da apresentação, a meta-informação para definição semântica do documento e as referências para conjuntos de elementos. De seguida é apresentado um exemplo de um cabeçalho no formato XMT-O.

```
<head>
   <layout metrics="pixel" type="xmt/xmt-basic-layout">
      <topLayout width="300" height="300" backgroundColor="branco">
         <region id="regiao_video" size="91 27"/>
         </region>
      </topLayout>
   </layout>
</head>
```

Código 4-3 Código Exemplo da Linguagem XMT-O

Em XMT-O, as composições possuem semântica de sincronização e é no corpo do documento XMT-O que se descreve a semântica temporal e sequencial da cena, onde várias composições podem ser definidas, através dos elementos *par*, *seq* e *excl* (tipo de semântica temporal

implementada - paralela, sequencial ou exclusiva). Ainda no corpo do documento, as características para apresentação dos objectos multimédia são especificadas nos próprios elementos, através dos seus atributos. Além das características para apresentação, as relações entre esses objectos também podem estar incluídas nos seus atributos. Dessa forma, além de composições com semântica de sincronização, estas relações podem ser estabelecidas através de eventos, especificados nesses atributos. De seguida é apresentado o código referente a um corpo associado ao cabeçalho acima descrito (Código 4-4).

```
<body>
   <par>
      <video src="video.mp4" region="regiao_video" begin="0s" dur="indefinite"/>
      <audio src="audio.mp4" begin="0s" dur="indefinite"/>
   </par>
</body>
```

Código 4-4 Código Exemplo da Linguagem XMT-O – Corpo e Descrição

Neste exemplo, estes dois elementos referidos formam um documento que descreve uma cena em formato XMT-O.

4.3.2.2 - SMIL

O SMIL [34] é uma linguagem normalizada pelo W3C [31] e foi desenvolvida desde início como uma linguagem para apresentações multimédia. O estudo do SMIL foi muito importante no contexto apresentado, porque apontou para a sua utilização, evidenciando as suas características adequadas ao sistema. O SMIL permite a gestão de apresentações de audiovisuais interactivas que integram fluxos de áudio, imagens e texto, ou qualquer outro tipo de objecto multimédia. Para isso, possui funções para a definição de sequências, duração, posição e visibilidade dos vários elementos (objectos) constituintes da cena multimédia, oferecendo a possibilidade de descrever o comportamento temporal da apresentação e descrever a localização espacial dos objectos no ecrã.

A segunda versão desta linguagem (SMIL 2.0) veio acrescentar dados novos importantes no crescimento desta linguagem, nomeadamente a introdução de conceitos como modularização e perfis, que permitem uma expansão da linguagem de um modo relevante. Assim, existem módulos que podem ser utilizados com elementos, atributos e valores de atributos adequados à descrição de uma aplicação particular. Existem vários módulos disponíveis neste contexto, como por exemplo, o módulo de ligação, o módulo de controlo de conteúdo e o módulo de

apresentação. Além disso, é possível nesta linguagem seleccionar um perfil que descreve o conjunto de módulos que estão a ser utilizados num determinado documento, com a função de compatibilizar esta descrição com a versão anterior (SMIL 1.0) e assegurar a compatibilidade com a semântica e conteúdos SMIL entre os vários módulos.

Nos códigos seguintes, são apresentados exemplos de descrição de cena em SMIL, com a possibilidade de interacção dos conteúdos.

```
<smil xmlns="http://www.w3.org/2005/SMIL20/Language">
   <head>
      <layout>
         <region id="geral" width="30" height="50"/>
         <region id="especifica" width="300" height="500"/>
      </layout>
   </head>
   <body>
      <par>
         <a href="video1.avi" target="especifica" accesskey="a">
         <video region="geral" src="video1.avi" begin="0s" dur="indefinite"/></a>
         <a href="video5.avi" target="especifica" accesskey="a">
         <video region="geral" src="video5.avi" begin="0s" dur="indefinite"/></a>
      </par>
   </body>
</smil>
```

Código 4-5 Código Exemplo da Linguagem SMIL – 1

No código acima são formadas duas regiões e apresentados dois vídeos numa linha de tempo infinita. No código abaixo, é criada uma região global e duas sub regiões. Nestas, são apresentadas duas imagens, uma que é apresentada dos zero aos seis segundos e outra que é apresentada do 1 aos 5 segundos.

```
<smil>
 <head>
  <layout>
   <root-layout width="300" height="200" background-color="white" />
   <region id="vim_icon" left="75" top="50" width="32" height="32" />
   <region id="soja_icon" left="150" top="50" width="100" height="30" />
  </layout>
 </head>
 <body>
  <seq>
   <img src="vim32x32.gif" alt="The vim icon" region="vim_icon" dur="6s" />
   <img src="madewithsoja.gif" alt="Made with SOJA" region="soja_icon" dur="4s" begin="1s" />
  </seq>
 </body>
</smil>
```

Código 4-6 Código Exemplo da Linguagem SMIL – 2

De seguida é apresentada a análise comparativa destas três linguagens.

4.3.3 - Análise Comparativa

Antes da análise comparativa das tecnologias apresentadas, torna-se necessário distinguir o conjunto de critérios que, ao momento da especificação da aplicação específica, se revelaram

preponderantes para o seu desenvolvimento. A enumeração destes critérios de comparação é importante para a decisão final na escolha da linguagem de descrição de cena, componente essencial no protótipo a desenvolver. Dos vários critérios possíveis, foram enumerados e assumidos apenas seis que melhor servem para a comparação a realizar, são eles:

1 - Conjunto de elementos adequados: Dando lugar à monitorização existirão elementos a monitorizar num determinado ambiente. Na linguagem de descrição terá de existir um conjunto de elementos, atributos e valores associados adequados à descrição e manuseamento dos objectos na cena. Se estes dados não estiverem definidos, terão de ser definidos fora da norma das linguagens.

2 - Semântica própria para representar alterações temporais: É necessário que a linguagem possua formas de descrição de alterações temporais na monitorização, assim sendo, é valorizado o modo como estas alterações são representadas para, da melhor forma, se traduzir todas as sequências temporais de monitorização possíveis neste tipo de aplicações. Seria conveniente que esta sequência fosse lógica e simples para melhor compreensão de autores e editores da aplicação referida.

3 - Semântica adequada para representar alterações espaciais: O modo como são expressas as transformações na cena também será um critério relevante neste contexto. Torna-se importante a existência de formas básicas para a transformação espacial dos elementos, nomeadamente na sua posição e tamanho, e uma semântica própria também na representação fácil e intuitiva destas alterações.

4 - Interpretação na edição de conteúdos: É importante para o autor/editor dos conteúdos ter uma noção da representação da cena apenas por inspecção do ficheiro que a compõem. À partida as descrições terão de ser exprimidas em modo texto para possibilitarem a fácil edição e deverão ter um grau de complexidade baixo para, dessa forma, serem intuitivas e facilmente entendidas na interacção humana.

5 - Baixo volume de informação: Para existir facilidade no envio de informação e baixo tempo de atraso entre alterações à monitorização, um dos critérios que podem ser definidos passa pela baixa complexidade e tamanho do ficheiro que contém a linguagem da descrição de cena. Na aplicação em causa exige-se uma transferência

semi-contínua de conteúdos que pode ter frequências bastante altas, o que coloca este factor como relevante neste ponto.

6 - Escalabilidade e Modularização: Como último critério e não menos importante, convém referir que é intenção no desenvolvimento desta aplicação, utilizar linguagens que possam ser aplicadas no contexto referido e que possam ser extensíveis, permitindo algumas modificações, adaptações ou a utilização de incrementos modulares adequados.

Definidos os critérios que servirão para a avaliação, pode-se definir também uma escala de valores que quantifique, de forma objectiva, as linguagens em análise. Sendo assim são utilizados valores de 1 a 3 que devem corresponder à seguinte interpretação:

1. Linguagem que não satisfaz o critério estabelecido.
2. Linguagem que satisfaz o critério estabelecido com necessidade de algumas alterações adicionais.
3. Linguagem que satisfaz o critério estabelecido sem necessidade de alterações adicionais.

Ao realizar este estudo, em primeira instância pôde-se verificar que o XML está a influenciar de várias formas o formato MPEG, formato importante na esfera das aplicações multimédia. Neste momento, tanto o formato BIFS (norma MPEG-4) como o formato BiM [42] (norma MPEG7) estão a ser complementados com esquemas de representação XML levando o MPEG a utilizar conceitos de algumas especificações como SMIL e X3D [43].

Numa análise mais pormenorizada, pode-se referir que a linguagem de descrição de cena BIFS numa fase anterior à conversão e compressão apresenta um conjunto vasto de elementos de desenho facial e de figuras geométricas, que não são necessários numa aplicação de monitorização no contexto apresentado. Esta linguagem possui também um conjunto variado de funções construtoras de gráficos 2D/3D, que são, no âmbito da análise, igualmente dispensáveis. Mas convém referir nesta altura que o principal factor que leva a rejeitar esta opção como válida no desenvolvimento da aplicação referida, está relacionada com o formato nativo da linguagem. Uma vez que a linguagem BIFS é convertida em formato binário, pode-se verificar que não permite a edição ou reedição directa dos conteúdos.

O MPEG redireccionou o formato desta linguagem para outro formato próprio de XML: o XMT que, na norma XMT-A apresenta a mesma forma da versão BIFS mas em formato texto, já permitindo ao autor/editor uma criação e alteração de conteúdos mais directa. Apesar desta vantagem note-se que continua a existir um conjunto de elementos não adequados, herdados das construções geométricas do BIFS. Na realidade, por representar directamente todos os elementos presentes na arquitectura MPEG-4, o XMT-A continua a apresentar-se como uma linguagem algo complexa e extensa para a edição fácil na criação e alteração de conteúdos.

Sendo uma linguagem de menor complexidade, o XMT-O apresenta-se como uma boa solução em relação ao XMT-A. Esta norma é baseada em SMIL e por isso apresenta um conjunto mais adequado de elementos. De qualquer forma, este formato apresenta alguma compatibilidade com o formato de representação da cena em árvore como o BIFS e o XMT-A e, com isto, algumas das características do SMIL são implementadas de forma parcial, como por exemplo na sincronização, nas transições e controlo de conteúdos ou até não implementadas como no caso do tratamento de eventos de entrada (por exemplo, do teclado).

O SMIL apresenta-se como uma linguagem para descrição de cenas de princípio simples e bastante eficiente. É de referir que esta é a linguagem que apresenta os elementos mais adequados ao sistema em desenvolvimento, principalmente porque é uma linguagem com uma modularização abrangente. Baseado neste conceito, pode-se optar por utilizar alguns módulos que demonstram uma boa adequabilidade para este tipo específico de aplicação em desenvolvimento, como é o caso do:

- Módulo de Ligação
- Módulo de Controlo de Conteúdo
- Módulo de Apresentação

Após o estabelecimento dos critérios de análise e após esta análise particular da adequabilidade destas linguagens, irá ser definido o grau de adequabilidade e a medida da integração destas linguagens nestes sistemas. Ao providenciar esse conjunto possível de critérios e ao cruzar esta informação com as características inerentes às linguagens em análise, é apontada uma solução que será a mais adequada no contexto explicado. Na Tabela 1 é

apresentado o conjunto de linguagens em análise e a sua correspondente adequabilidade aos critérios formulados utilizando a escala de valores estabelecida (de 1 a 3).

Linguagens	Critério 1	Critério 2	Critério 3	Critério 4	Critério 5	Critério 6
XMT-A	1	2	3	1	3	1
XMT-O	2	3	3	3	3	3
SMIL 2.0	3	3	3	3	3	3

Tabela 1 Análise Comparativa da Adequabilidade das Linguagens Submetidas

Desta tabela sobressai que as linguagens XMT-O e o SMIL se apresentam como boas candidatas à aplicação desenvolvida. Note-se a diferença existente entre o formato XMT-A e XMT-O baseada em primeira instância no critério de adequação de elementos (critério 1), onde se verifica que o XMT-O oferece um conjunto de elementos mais adequados do que o XMT-A. Em segunda instância também se verificam diferenças notórias na facilidade de interpretação na edição de conteúdos e consequente complexidade (critério 4), onde o XMT-A revelou ser mais complexo do que o formato XMT-O. Quanto ao critério 6 pode-se verificar que existe uma discrepância entre o XMT-A e os outros dois formatos. Isto deve-se à estrita ligação do XMT-A com o formato binário, não permitindo qualquer modularização ou capacidade de adaptação por parte deste formato à aplicação em desenvolvimento. Sendo assim, o XMT-O e o SMIL são apresentados como formatos válidos, embora com algumas pequenas diferenças entre eles. Ambos diferem no conjunto de elementos que disponibilizam (critério 1), sendo o SMIL o formato mais adequado, uma vez que, para além dos elementos principais (objectos multimédia) e seus atributos, este permite um conjunto vasto de possibilidades de interacção com o conteúdo e transições adequadas ao contexto em causa. Sob outra perspectiva, no critério 6 também pode-se observar algumas diferenças. Neste critério, o SMIL apresenta-se com um conjunto vasto de módulos que permitem adaptações a vários tipos de sistemas. Alguns destes revelaram ser bastante úteis para a aplicação, nomeadamente os módulos de ligação e apresentação. Apesar de alguns destes conceitos também estarem presentes no XMT-O, o SMIL apresenta uma maior adequabilidade à aplicação a desenvolver. Em suma, o SMIL possuí vantagens adicionais em relação ao XMT-O, que passam pela existência de elementos primários adequados (com os seus atributos respectivos) e pela capacidade de modularização disponível, oferecendo um conjunto adicional de elementos

que permitem descrever melhor a cena da aplicação em causa e permitindo uma futura escalabilidade do sistema.

O seguinte código apresenta um exemplo simples de uma descrição de cena, onde os objectos centrais da cena são dois objectos vídeo que representam o fluxo das imagens do ambiente de trabalho de cada computador a monitorizar.

```
<smil>
   <head>
      <layout >
         <root-layout width="320" height="240"/>
         <region id="janela1" left="170" top="110"/>
         <region id="janela2" left="50" top="50"/>
      </layout>
   </head>
   <body>
      <par>
         <video src="comp1.avi" begin="5s" region="janela1"/>
         <video src="comp5.avi" begin="2s" region="janela2"/>
      </par>
   </body>
</smil>
```

Código 4-7 Linguagem de Descrição de Cena - SMIL

No formato da linguagem SMIL, depois de definidos os objectos e seus atributos, é possível definir ainda interacções entre eles e transformações espaciais e temporais adequadas. Estas transformações poderão ser definidas no futuro para traduzir níveis de importância de monitorização, de acordo com critérios definidos antes do início da monitorização, ou ao longo do processo de monitorização, de acordo com valores de variáveis que se pretendam monitorar.

Tal como é indicado na secção do resultado da análise, a linguagem SMIL foi a solução que se demonstrou mais adequada aos critérios exigidos. Tendo por base este resultado, foram formuladas e testadas várias descrições de cena no protótipo. Assim, a informação de configuração de ambiente foi definida em XML simples, da seguinte forma:

```
<ambconfig>

      <monitor colunas="4" linhas="3"/>

      <src janela="1" linha="1" coluna="1" />
      <src janela="2" linha="1" coluna="2" />
      <src janela="3" linha="1" coluna="3" />
      <src janela="4" linha="1" coluna="4" />
      <src janela="5" linha="2" coluna="1" />
           (...)
      <src janela="12" linha="3" coluna="4" />

      <src computador="1" aluno="Antonio Coelho" ip="192.168.0.1"/>
      <src computador="2" aluno="Antonio Coelho" ip="192.168.0.2"/>
      <src computador="3" aluno="Antonio Coelho" ip="192.168.0.3"/>
      <src computador="4" aluno="Antonio Coelho" ip="192.168.0.10"/>
      <src computador="5" aluno="Antonio Coelho" ip="192.168.0.11"/>
      <src computador="6" aluno="Antonio Coelho" ip="192.168.0.12"/>

      <meta var=”proc_perm”>
         <processo>explorer.exe<\processo>
         <processo>svchost.exe<\processo>
      <meta var=”proc_nperm”>
         <processo>messenger.exe<\processo>
         <processo>emule.exe<\processo>

</ambconfig>
```

Código 4-8 Informação de Configuração de Ambiente (XML)

A descrição de cena, por sua vez é representada num esquema SMIL, apresentada no Código 4-9.

```
<smil>
   <sdesc>
      <seq repeatCount="indefinite">

      <video computador="1" janela="1" begin="0" dur="2" />
      <video computador="1" janela="2" begin="2" dur="2" />
      <video computador="1" janela="3" begin="4" dur="2" />
      <video computador="1" janela="4" begin="7" dur="2" />
      <video computador="1" janela="5" begin="8" dur="2" />
      <video computador="1" janela="6" begin="10" dur="2" />
      <video computador="1" janela="7" begin="12" dur="2" />

                                      (...)

      <video computador="5" janela="9" begin="4" dur="2" />
      <video computador="5" janela="10" begin="6" dur="2" />

      <video computador="6" janela="1" begin="10" dur="2" />
      <video computador="6" janela="2" begin="12" dur="2" />
      <video computador="6" janela="3" begin="14" dur="2" />
      <video computador="6" janela="4" begin="16" dur="2" />
      <video computador="6" janela="5" begin="18" dur="2" />
      <video computador="6" janela="6" begin="0" dur="2" />
      <video computador="6" janela="7" begin="2" dur="2" />
      <video computador="6" janela="8" begin="4" dur="2" />
      <video computador="6" janela="9" begin="6" dur="2" />
      <video computador="6" janela="10" begin="8" dur="2" />

      </seq>
   </sdesc>
</smil>
```

Código 4-9 Informação de Descrição de Cena (SMIL)

É de referir ainda que estas duas informações, a informação de Descrição de Ambiente e Descrição de Cena poderão ser utilizadas no mesmo código XML. Esta abordagem foi adoptada para a concretização do protótipo sendo esta informação designada como Linguagem de Descrição Geral e pode ser vista no Código 5-1, Tópico 5.1.

4.4 **- Esquemas de Validação XML**

Para uma consistente formação destas linguagens em formato XML foram testados alguns esquemas de validação de gramática estrutural para este tipo de formatos, como é o caso do XML Schema [44], e validações por software, caso do Schematron [45]. A título de exemplo no código abaixo pode ser visto uma parte de um XML Schema utilizado para validar a estrutura da linguagem de descrição de cena definida nos tópicos anteriores.

```
<?xml version="1.0"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
<xs:element name="ambconfig">
   <xs:complexType>
      <xs:sequence>
   <xs:element name="monitor">
   <xs:complexType>
      <xs:sequence>
      <xs:element name="colunas">
         <xs:simpleType>
            <xs:restriction base="xs:integer"><xs:pattern value="[0-9]"/></xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="linhas">
         <xs:simpleType>
            <xs:restriction base="xs:integer"><xs:pattern value="[0-9]"/></xs:restriction>
         </xs:simpleType>
      </xs:element>
      </xs:sequence>
   </xs:complexType>
   </xs:element>
   <xs:element name="src">
   <xs:complexType>
      <xs:sequence>
      <xs:element name="janela">
         <xs:simpleType>
            <xs:restriction base="xs:integer"><xs:pattern value="[0-9]"/></xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="linha">
         <xs:simpleType>
            <xs:restriction base="xs:integer"><xs:pattern value="[0-9]"/></xs:restriction>
         </xs:simpleType>
      </xs:element>
      <xs:element name="coluna">
         <xs:simpleType>
            <xs:restriction base="xs:integer"><xs:pattern value="[0-9]"/></xs:restriction>
         </xs:simpleType>
      </xs:element>
      </xs:sequence>
   </xs:complexType>

      (…)

</xs:element>
```

Código 4-10 Código em XML Schema - Validação da Informação de Configuração de Ambiente

Constituiu sempre uma preocupação, que todas as linguagens estivessem no formato correcto e de acordo as normas definidas. Assim, este procedimento levou a que se pudesse construir de forma fiel várias possibilidades de descrições de cena para integrar no protótipo construído.

4.5 **- Windows Management Instrumentation**

O WMI - Windows Management Instrumentation [46] é uma especificação Microsoft que visa estabelecer funções de acesso e partilha de informações de gestão de sistemas, utilizando como base uma rede de computadores. O WMI inclui um repositório de objectos, que é a base de dados das definições do objecto e o gestor de objectos CIM (*Common Information Model*) [47], que trata da recolha e manipulação de objectos no repositório e reúne informações dos fornecedores de WMI.

O WMI é a base de ferramentas de gestão de computadores como é o caso do Microsoft Systems Management Server. O WMI pode também ser utilizado com sistemas de programação (tal como o Perl) ou de processamento de scripts (tal como PHP ou Windows Script Host), para assim obter detalhes e interagir com algumas propriedades e definições de um computador local ou remoto. Desta forma, é possível por exemplo reiniciar um computador remoto para aplicar alterações de definições ou detectar hardware novo, visualizar o nome do computador e informações sobre o domínio para outros computadores na rede, ou verificar e alterar definições importantes que influenciam o estado actual do sistema operativo como os seus processos, memória ou prioridades no processamento.

Existem outro tipo de ferramentas cujo objectivo também é a gestão de propriedades em sistemas operativos, como é o caso do protocolo SMNP – Simple Network Management Protocol [48] que não poderia deixar de ser mencionado neste trabalho, tendo em conta a sua importância nesta área e a sua característica importante de interoperabilidade entre vários sistemas operativos. Todavia, uma vez que apenas se pretende gerir sistemas operativos Microsoft e uma vez que o WMI se apresentou como uma ferramenta versátil e adequada ao sistema a desenvolver, o WMI foi a escolha para a comunicação de listas e acções sobre processos para o protótipo. No Anexo 8.3 pode ser visto o ecrã de uma ferramenta de teste utilizada no desenvolvimento de funções WMI: Windows Management Instrumentation Tester.

Capítulo 5 - Implementação do Protótipo de Monitorização

Neste capítulo, é apresentado todo o processo que envolveu a concretização de um protótipo, respondendo ao método de monitorização apresentado no Capítulo 2. As tecnologias abordadas no capítulo anterior são agora aplicadas no *Agente de Configuração e Apresentação*, no *Agente de Monitorização* e no *Agente de Prova*.

O desenvolvimento global deste sistema envolveu três etapas principais: a especificação, o desenvolvimento/implementação do protótipo e os testes finais. A etapa do desenvolvimento/implementação do protótipo foi planeada com a constituição de fases, são elas:

- Fase 1 - Construção do Agente de Monitorização
- Fase 2 – Construção do Agente de Prova
- Fase 3 – Testes Intermédios
- Fase 4 - Construção do Agente de Configuração e Apresentação
- Fase 5 – Testes Globais

Na Tabela 2 é apresentado o esquema temporal de cada uma destas fases no desenvolvimento do protótipo.

Fases / Semanas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28
Fase 1																												
Fase 2																												
Fase 3																												
Fase 4																												
Fase 5																												

Tabela 2 Cronograma das Fases de Desenvolvimento/Implementação do Protótipo

Nesta tabela pode ser observado que as duas primeiras fases correspondentes à construção do *Agente de Monitorização* e do *Agente de Prova* foram desenvolvidas logo de início (com a inclusão do *Agente de Configuração e Apresentação* no *Agente de Monitorização*), até à fase 3 de

testes intermédios. Depois da fase 3, iniciou-se o desenvolvimento do *Agente de Configuração e Apresentação* fora do *Agente de Monitorização* até aos testes finais globais. Note-se ainda que, durante o desenvolvimento do *Agente de Configuração e Apresentação*, foi necessário alterar os dois Agentes construídos nas fases antecedentes.

No início da construção dos Agentes pertencentes ao sistema, foram adoptadas algumas simplificações ao seu esquema. Assim, optou-se por colocar em funcionamento logo de início, a dinâmica *Agente de Monitorização – Agente de Prova*. De início todas as funções do *Agente de Configuração e Apresentação* (e portanto todos os seus blocos constituintes) foram incluídos no *Agente de Monitorização*. Depois de acertada a dinâmica entre o *Agente de Configuração e Apresentação* e o *Agente de Monitorização* na mesma máquina, e da realização dos testes intermédios descritos nas fases de implementação, estes dois blocos foram separados. Esta opção, permitiu desenvolver o sistema, por forma a incluir vantagens adicionais não previstas na fase de especificação, como por exemplo, o caso específico do sistema funcionar com o *Agente de Configuração e Apresentação* completamente integrado no *Agente de Monitorização* (o que nalguns casos pode constituir uma vantagem de economia de recursos), ou no caso de permitir a existência de vários *Agentes de Configuração e Apresentação*, o que pode igualmente ser vantajoso no caso de termos vários Examinadores na prova (este modo não foi totalmente conseguido, uma vez que são necessárias algumas considerações adicionais, como a comunicação entre cada um destes Agentes de função semelhante).

5.1 **- Agente de Monitorização – Agente de Prova**

Tal como já foi referido, optou-se por, numa primeira fase do desenvolvimento do protótipo, implementar apenas o *Agente de Monitorização* e o *Agente de Prova*. Isto seria conseguido através da colocação das funções do *Agente de Configuração e Apresentação* no *Agente de Monitorização*. Este *Agente de Configuração e Apresentação* tem algumas funções importantes, nomeadamente o envio do Fluxo Configurador Principal, a recepção do fluxo de informação de Processos e Ecrãs e o estabelecimento dos fluxos actuadores de Ecrã para depois receber o Fluxo de Monitorização de Ecrã, como se pode observar na Figura 5-1. Recordando este esquema já

descrito na fase de especificação, o fluxo Configurador inclui duas informações necessárias para a inicialização do sistema: A configuração de Ambiente e a Descrição de Cena.

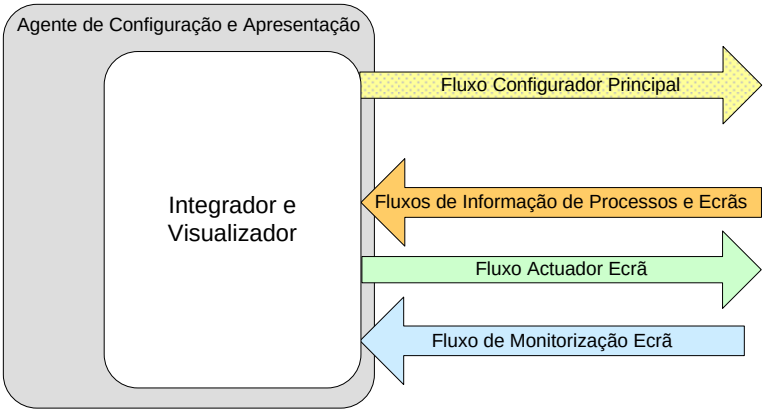


Figura 5-1 Configuração de Ambiente e Descrição de Cena

Para a implementação destas funções no *Agente de Monitorização*, e de acordo com os estudos realizados às tecnologias existentes, foi utilizada a linguagem XML para representar estas duas linguagens. Num ficheiro presente no disco rígido, já simulando a presença de parte do bloco integrador e visualizador do *Agente de Configuração e Apresentação*, a primeira informação que define o espaço de monitorização, a informação de configuração de ambiente, e a informação de descrição de cena, foram agrupadas num único ficheiro no formato SMIL, doravante designada como Linguagem de Descrição Geral e no formato conforme o excerto apresentado no Código 5-1.

```
<smil>
   <head>
      <monitor linhas="3" colunas="4" />

      <src janela="1" linha="1" coluna="1" />
      <src janela="2" linha="1" coluna="2" />
      <src janela="3" linha="1" coluna="3" />
      (…)
      <src computador="1" aluno="Pedro Pinto" ip="192.168.0.1"/>
      <src computador="2" aluno="Antonio Coelho" ip="192.168.0.2"/>

      <meta var=”proc_perm”>
         <processo>explorer.exe<\processo>
         <processo>svchost.exe<\processo>
      <meta var=”proc_nperm”>
         <processo>messenger.exe<\processo>
         <processo>emule.exe<\processo>
   </head>
   <body>
      <seq repeatCount="indefinite">
         <video computador="1" janela="1" begin="0" dur="2" />
         <video computador="1" janela="2" begin="2" dur="2" />
            (…)
         <video computador="6" janela="10" begin="8" dur="2" />
      </seq>
   </body>
</smil>
```

Código 5-1 Linguagem de Descrição Geral – SMIL

Produzidas as duas informações referidas, estas são encaminhadas para o bloco Monitorizador, através do bloco de Configuração e Apresentação, como se pode ver na Figura 5-2.

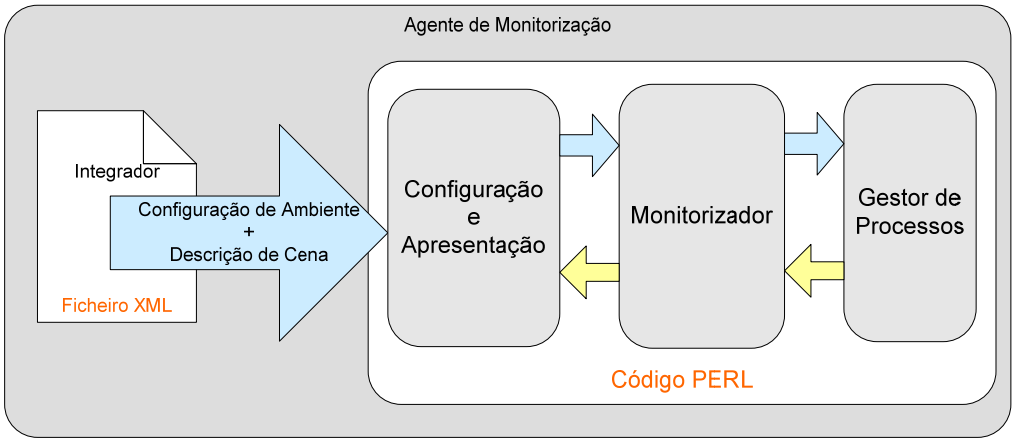


Figura 5-2 Agente de Monitorização – Supressão do Agente de Configuração e Apresentação 1

Também pode ser observado que os três blocos que constituem o Agente Monitorizador (o bloco de Configuração e Apresentação, o Monitorizador e o Gestor de Processos) foram implementados num único programa com código PERL, como se apresenta no Código 5-2, no Código 5-3 e no Código 5-4, apresentado de forma completa no Capítulo 8.

```
$eventos=0;
print " - > Script Progress 4 - Getting XML Values\n";
while (my($chave1, $valor1) = each(%$scene)) {
    print (" 1 Chave: $chave1\t\tValor: $valor1\n");
    if ($chave1 eq "body") {
       $body = $valor1;
       foreach $tseq (@$body){
          while(my($chave2, $valor2) = each (%$tseq)){
             print (" 2 Chave: $chave2\t\tValor: $valor2\n");
             while(my($chave3, $valor3) = each (%$valor2)){
                print (" 3 Chave: $chave3\t\tValor: $valor3\n");
                if ($chave3 eq "repeatCount"){
                        if($valor3 eq "indefinite"){$repeat=0;}
                        else {$repeat=$valor3;}
```

Código 5-2 Código Agente de Monitorização – Captura de Valores na Descrição de Cena

```
for ($j==1; $j<=$eventos; $j++) {
    #print "----- $begin[$j]+$dur[$j]\n";
    if (($begin[$j]+$dur[$j])>$scene_time) {
       $scene_time=($begin[$j]+$dur[$j]);
    }
}
```

Código 5-3 Código Agente de Monitorização – Construção de Eventos de Monitorização

```
if ($begin[$mark]+$dur[$mark]==$clock || $begin[$mark]+$dur[$mark]==$multiplo ||
($begin[$mark]+$dur[$mark])%$scene_time==$multiplo) {
         print "Stop\tcomp[$computador[$mark]] on window[$janela[$mark]] - close
at=$begin[$mark]+$dur[$mark]\n";
         }
```

Código 5-4 Código Agente de Monitorização – Envio de Mensagens para Agente de Prova

Por outro lado o bloco Integrador e Visualizador recebe também o fluxo de informação de Processos e Ecrãs. As funções desta parte do bloco também foram implementadas no *Agente de Monitorização* com recurso a um programa em código PERL (Figura 5-3). A partir deste ponto passamos a ter o *Agente de Configuração e Apresentação* completamente emulado dentro do *Agente de Monitorização*.

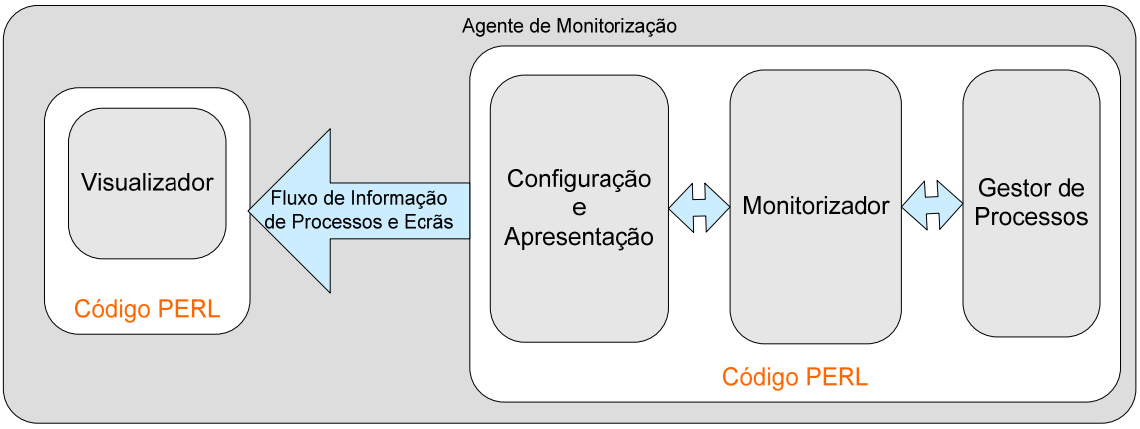


Figura 5-3 Agente de Monitorização – Supressão do Agente de Configuração e Apresentação 2

De seguida, procedeu-se à construção do *Agente de Prova*. Inicialmente, apenas foi programada a construção de um *Agente de Prova*. A implementação deste agente obedeceu a

uma condição importante: a exigência de alterações mínimas na configuração da máquina em causa. Sendo um computador de características comuns, onde um determinado indivíduo a examinar realiza a sua prova, é de todo o interesse que a sua configuração seja leve, fácil e rápida de realizar. A Figura 5-4 pretende recordar que o *Agente de Prova* é composto por duas interfaces: a interface monitorização de processos e o bloco de monitorização de ecrãs.

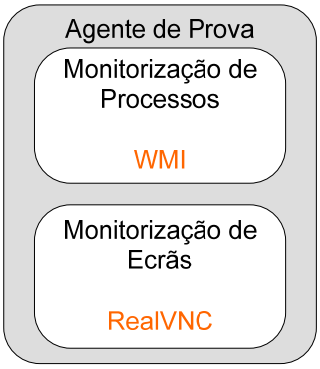


Figura 5-4 Esquema do Agente de Prova

A implementação do bloco de monitorização de processos, é conseguida através da utilização da interface genérica já descrita: o WMI. Conforme pode ser visto na Figura 5-5, toda a interacção entre o Gestor de Processos do lado do *Agente de Monitorização* e o bloco de Monitorização de Processos no *Agente de Prova* é integralmente realizada recorrendo à interface WMI.

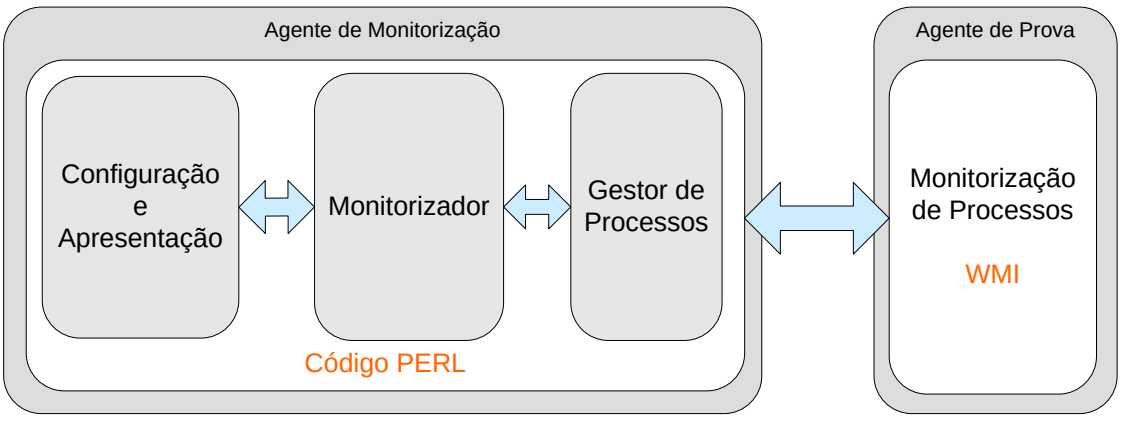


Figura 5-5 Interacção entre Gestor de Processos e Interface de Monitorização de Processos

A monitorização dos processos recorre a esta interface para configurar a geração de alertas a enviar ao Gestor de Processos. Estes alertas são disparados quando são realizadas acções

relevantes (estabelecidas na configuração). Assim, são criados eventos em WMI no bloco Configuração que tem como objectivo recolher os dados necessários no *Agente de Prova* (é apresentado abaixo um exemplo no Código 5-5 presente em [49], cuja função é pedir a lista de processos de uma determinada máquina na rede).

```
#   ListProc.pl

use Win32::OLE qw( in );
use Win32::OLE::Variant;

$Machine = "\\\\.";
$Machine = shift @ARGV if( $ARGV[0] =~ /^\\\\/ );

# WMI Win32_Process class
$CLASS = "winmgmts:{impersonationLevel=impersonate}$Machine\\Root\\cimv2";
$WMI = Win32::OLE->GetObject( $CLASS ) || die;
foreach my $Proc ( sort {lc $a->{Name} cmp lc $b->{Name}} in( $WMI->InstancesOf( "Win32_Process"
) ) )
{
  printf( "% 5d) %s ", $Proc->{ProcessID}, "\u$Proc->{Name}" );
  print "( $Proc->{ExecutablePath} )" if( "" ne $Proc->{ExecutablePath} );
  print "\n";
}
```

Código 5-5 Diagrama de Blocos – Gestor de Processos

Desta forma a comunicação é completamente transparente, de acordo da tecnologia WMI. No código seguinte (Código 5-6), pode-se ver algumas das funções que são possíveis com o módulo Win32::Process.

```
Win32::Process::Create($obj,$appname,$cmdline,$iflags,$cflags,$curdir)
Creates a new process.
    Args:
        $obj            container for process object
        $appname        full path name of executable module
        $cmdline        command line args
        $iflags         flag: inherit calling processes handles or not
        $cflags         flags for creation (see exported vars below)
        $curdir         working dir of new process
Win32::Process::KillProcess($pid, $exitcode)
Terminates any process identified by $pid. $exitcode will be set to the exit code of the process.
$ProcessObj->Suspend()
Suspend the process associated with the $ProcessObj.
$ProcessObj->Resume()
Resume a suspended process.
$ProcessObj->Kill( $exitcode )
Kill the associated process, have it terminate with exit code $ExitCode.
$ProcessObj->GetPriorityClass($class)
Get the priority class of the process.
$ProcessObj->SetPriorityClass( $class )
Set the priority class of the process (see exported values below for options).
$ProcessObj->GetProcessAffinitymask( $processAffinityMask, $systemAffinitymask)
Get the process affinity mask. This is a bitvector in which each bit represents the processors
that a process is allowed to run on.
$ProcessObj->SetProcessAffinitymask( $processAffinityMask )
Set the process affinity mask. Only available on Windows NT.
$ProcessObj->GetExitCode( $exitcode )
Retrieve the exitcode of the process.
$ProcessObj->Wait($timeout)
Wait for the process to die. $timeout should be specified in milliseconds. To wait forever,
specify the constant INFINITE.
$ProcessObj->GetProcessID()
Returns the Process ID.
```

Código 5-6 Manipulação de Processos através do WMI

Quanto ao bloco de Monitorização de Ecrãs, esta é implementada com recurso a um servidor da RealVNC. Este, quando solicitado, envia o fluxo do ecrã visualizado, tal como foi descrito atrás. Para que o *Agente de Monitorização* possa receber esse fluxo precisa de ter instalado um Cliente VNC no bloco Visualizador. Todo o conjunto de blocos que foi desenvolvido é apresentado na Figura 5-6. O esquema inclui a tecnologia associada a cada bloco.

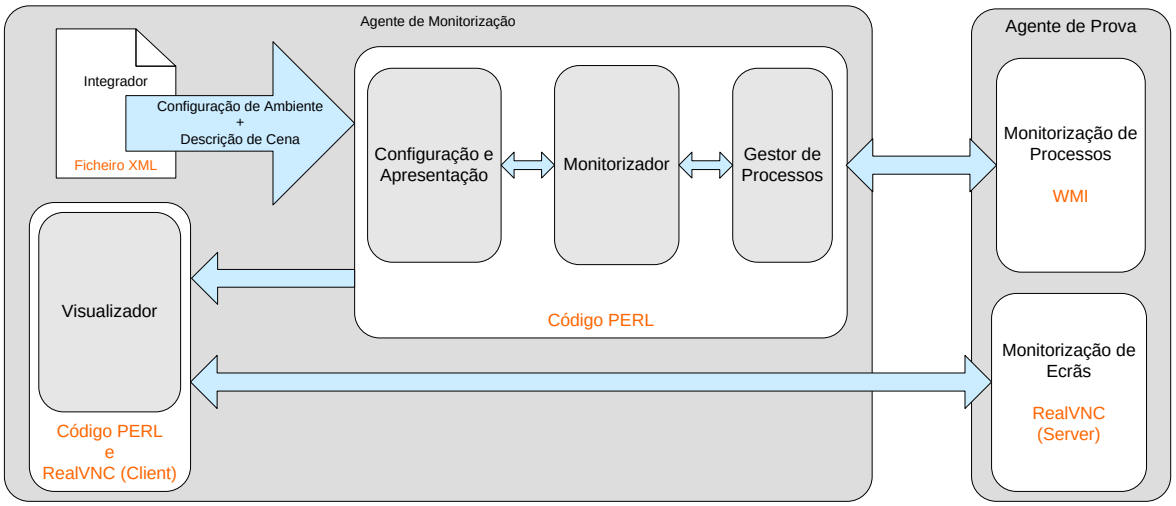


Figura 5-6 Diagrama de Blocos da Entidade Agente de Monitorização

De seguida é apresentado o desenvolvimento do *Agente de Configuração e Apresentação*.

5.2 - Agente de Configuração e Apresentação

O *Agente de Configuração e Apresentação* é um agente de construção simples deste sistema. Numa primeira fase este agente foi embebido no *Agente de Monitorização*. Ele é composto por um conjunto de blocos que foram construídos com recurso à programação em Perl, ao formato PHP (ambos já referidos na secção 4.1 deste documento) e ao software VNC (na secção 4.2).

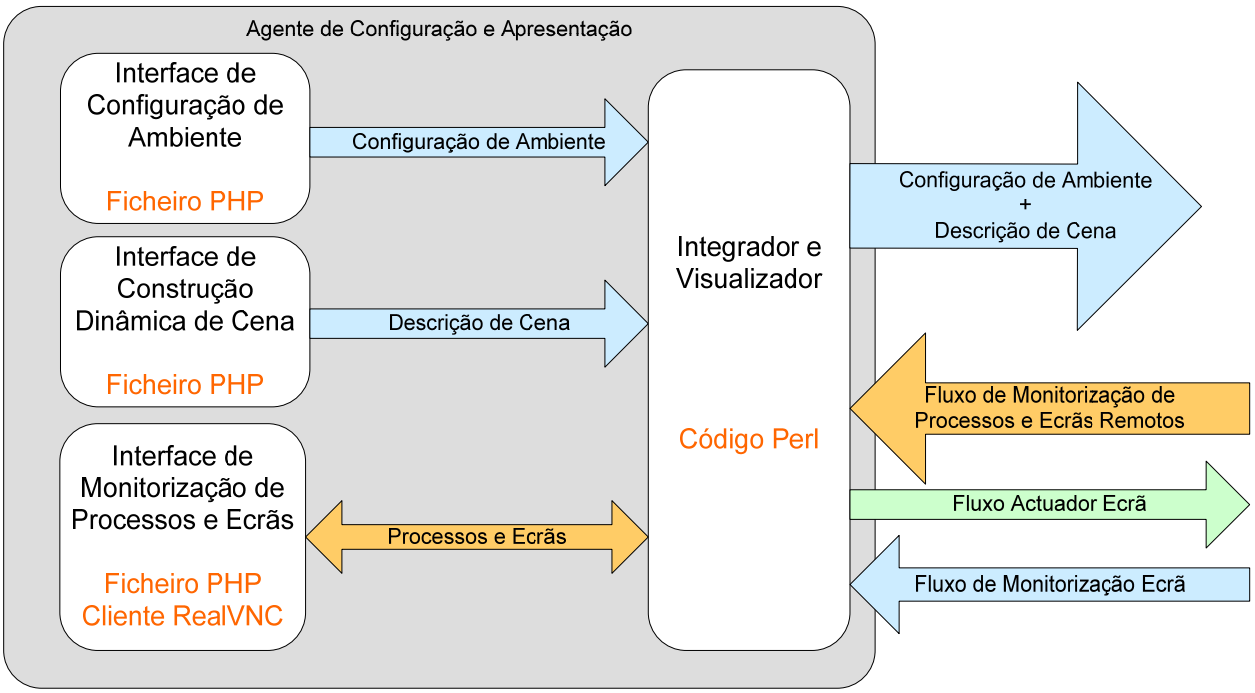


Figura 5-7 Esquema do Agente de Configuração e Apresentação

O bloco Interface de Configuração de Ambiente e o bloco de Construção Dinâmica de Cena são realizados através de um ficheiro em PHP apresentado num Browser. A apresentação do PHP pode ser visto na Figura 5-8 (o código em PHP que produz este resultado no browser é apresentado no Anexo 8.5)

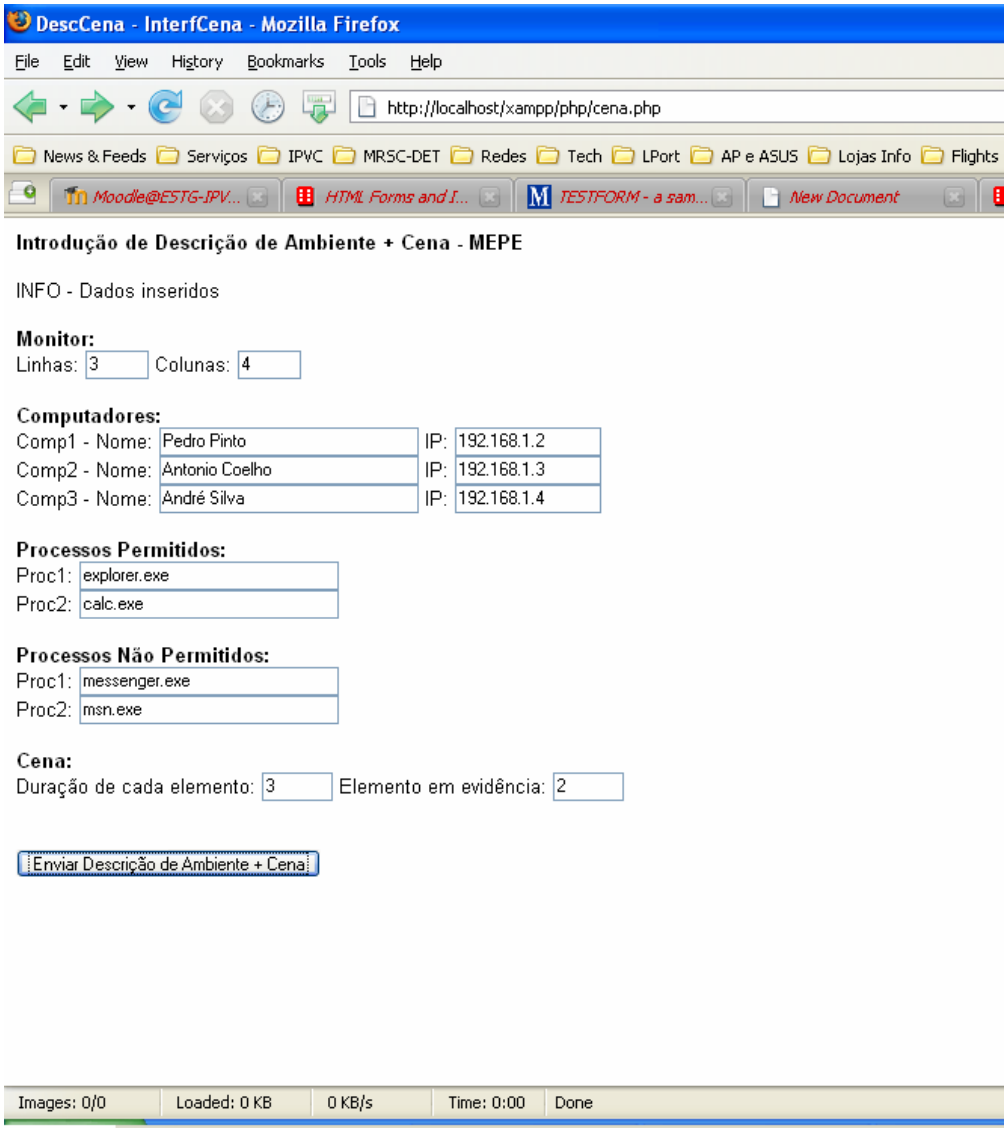


Figura 5-8 Interface PHP num Browser para Construção da Descrição de Ambiente e Cena

No browser, o utilizador (examinador) criará implicitamente a duas informações (Descrição de Ambiente e de Cena) em formato SMIL. No caso da página acima de PHP foi criado de forma automática o Código 5-7 que se apresenta em seguida (apresentado em formato completo no Anexo 8.6).

```
<smil><head>
   <monitor linhas="3" colunas="4"/>

   <src janela="1" linha="1" coluna="1"/>
   <src janela="2" linha="1" coluna="2"/>
   <src janela="3" linha="1" coluna="3"/>
   <src janela="4" linha="1" coluna="4"/>
   <src janela="5" linha="2" coluna="1"/>
   <src janela="6" linha="2" coluna="2"/>
   <src janela="7" linha="2" coluna="3"/>
   <src janela="8" linha="2" coluna="4"/>
   <src janela="9" linha="3" coluna="1"/>
   <src janela="10" linha="3" coluna="2"/>
   <src janela="11" linha="3" coluna="3"/>
   <src janela="12" linha="3" coluna="4"/>

   <src computador="1" formando="Pedro Pinto" ip="192.168.1.2"/>
   <src computador="2" formando="Antonio Coelho" ip="192.168.1.3"/>
   <src computador="3" formando="André Silva" ip="192.168.1.4"/>

   <meta var="proc_perm"/>
   <processo> explorer.exe</processo>
   <processo> calc.exe</processo>
   <meta var="proc_nperm"/>
   <processo> messenger.exe</processo>
   <processo> msn.exe</processo>

</head>
<body>
   <seq repeatCount="indefinite">
   <video computador="1" janela"1" begin="0" dur="3"/>
   <video computador="1" janela"2" begin="3" dur="3"/>
   <video computador="1" janela"3" begin="6" dur="3"/>
   <video computador="1" janela"4" begin="9" dur="3"/>
   <video computador="1" janela"5" begin="12" dur="3"/>
   <video computador="1" janela"6" begin="15" dur="3"/>
   <video computador="1" janela"7" begin="18" dur="3"/>

           (...)

   <video computador="3" janela"1" begin="0" dur="3"/>
   <video computador="3" janela"2" begin="3" dur="3"/>
   <video computador="3" janela"3" begin="6" dur="3"/>
   <video computador="3" janela"4" begin="9" dur="3"/>
   <video computador="3" janela"5" begin="12" dur="3"/>
   <video computador="3" janela"6" begin="15" dur="3"/>
   <video computador="3" janela"7" begin="18" dur="3"/>
   <video computador="3" janela"8" begin="21" dur="3"/>
   <video computador="3" janela"9" begin="24" dur="3"/>
   <video computador="3" janela"10" begin="27" dur="3"/>
   <video computador="3" janela"11" begin="30" dur="3"/>
   <video computador="3" janela"12" begin="33" dur="3"/>
   </seq>
</body></smil>
```

Código 5-7 Código gerado na Interface de Construção de Ambiente e Cena

Estes fluxos são enviados para um bloco integrador e visualizador, um bloco em PERL que procede à integração destes fluxos e envio para o *Agente de Monitorização*.

É também o bloco Integrador e Visualizador que processa o fluxo de monitorização de processos e ecrãs remotos. Mediante a configuração existente, este bloco envia os fluxos actuadores de ecrã para os *Agentes de Prova* que depois originam os fluxos de monitorização de ecrãs em sentido inverso, tal como já foi referido na especificação. Isto é realizado através do código PERL programado no bloco Integrador e Visualizador. A informação relevante e o fluxo de monitorização de ecrã, que representa um fluxo VNC vão ser apresentados através da interface de Processos e ecrãs: os processos num ficheiro PHP, e os ecrãs através de um cliente RealVNC.

No projecto de toda a arquitectura de funcionamento deste agente, foram delineados todos os processos e funções para que fosse possível separar em duas entidades distintas. Assim, num plano futuro e com alterações mínimas ao protótipo desenvolvido, pode-se admitir a existência de um Agente de Configuração (apenas com funções de configuração do sistema) e um, ou vários, Agentes de Apresentação (apenas com funções de estabelecimento de fluxos e de apresentação de ecrãs).

5.3 – Estrutura de Suporte – Topologia de Rede

O ambiente para a implementação e teste desta tecnologia exige algumas condições. À partida, e como já foi referido, realizou-se uma fase de testes apenas com o *Agente de Monitorização* e o *Agente de Prova*.

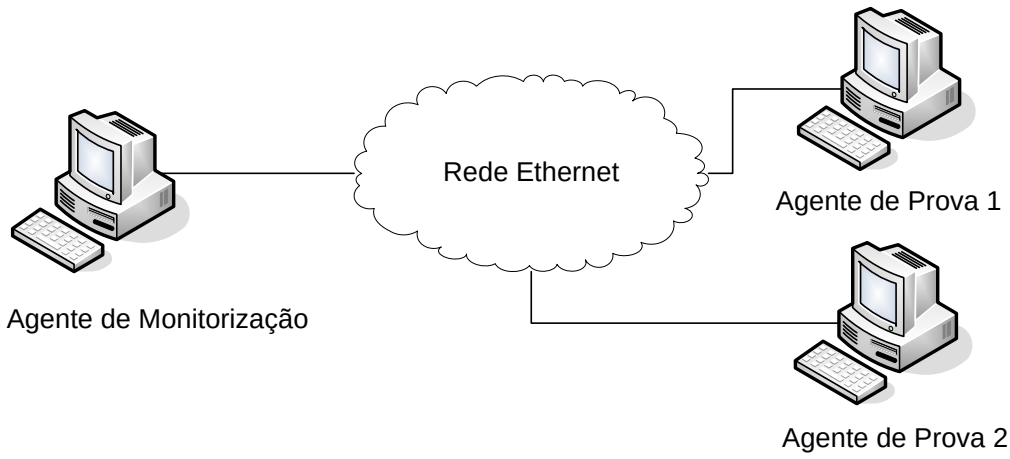


Figura 5-9 Estrutura Lógica de Rede – Agente de Monitorização e Agente de Prova

A estrutura lógica de rede criada para o efeito é apresentada na Figura 5-9 e a estrutura física é apresentada na Figura 5-10. Todos os computadores utilizados na fase de implementação do protótipo possuíam características semelhantes de entre as quais se destacam os seguintes elementos: CPU: 2.4GHz, RAM: 512MB, HD: 80G, DVD-ROM.

De início apenas foram montados dois computadores, o *Agente de Monitorização* e o *Agente de Prova* 1. Os dois computadores foram interligados com recurso a um sistema de rede Ethernet [50], recorrendo a um *Switch* (Comutador) não configurável, simples, com 8 portas de 100Mbps (RJ-45).

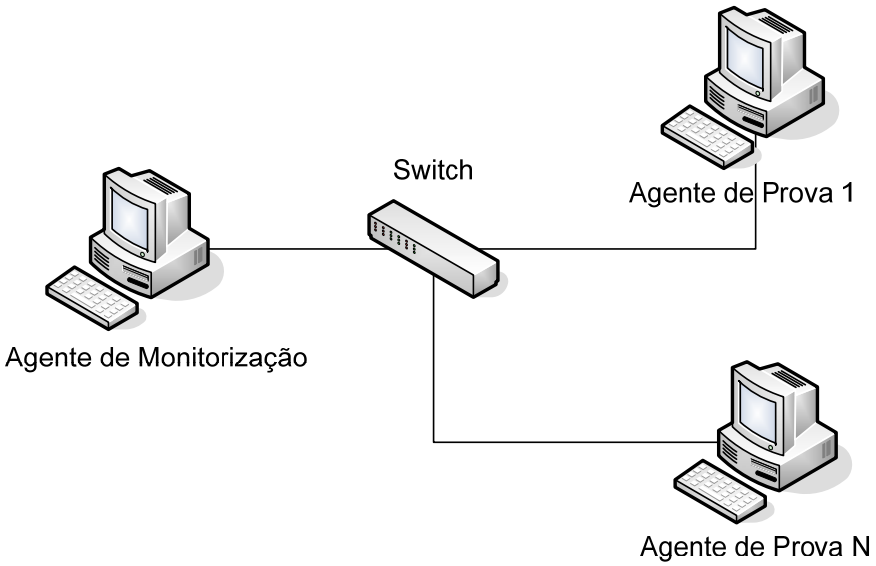


Figura 5-10 Estrutura Física de Rede – Agente de Monitorização e Agente de Prova

No *Agente de Monitorização* foram instaladas as seguintes aplicações:

- Windows XP – SP2 [Version 5.1.2600]
- XAMPP 1.6.4
- ActivePerl 5.8.8.822

No *Agente de Prova* foram instaladas ou configuradas as seguintes aplicações:

- Windows XP – SP2 [Version 5.1.2600]
- Componente WMI SNMP Provider

Alguns testes foram realizados com esta topologia, e foram obtidos os resultados previstos na comunicação entre estes dois agentes. Numa fase seguinte, foi considerado um segundo *Agente de Prova*, conforme é apresentado na Figura 5-10. A intenção, neste ponto, era verificar se existiriam problemas gerados por conflitos, quer no endereçamento da interface de WMI de cada um deles, ou relativo ao endereçamento e estabelecimento de fluxos entre cliente e servidor RealVNC. Além disso, neste ponto foi interessante analisar quais eram as exigências da solução construída relativamente a duas características específicas, nomeadamente a largura de banda e os atrasos verificados na rede. Num ambiente normal poder-se-ia ter vinte, trinta ou cinquenta alunos, por isso, para testar a escalabilidade desta solução, realizaram-se alguns testes e

extrapolaram-se resultados que permitiram prever o funcionamento do protótipo em larga escala.

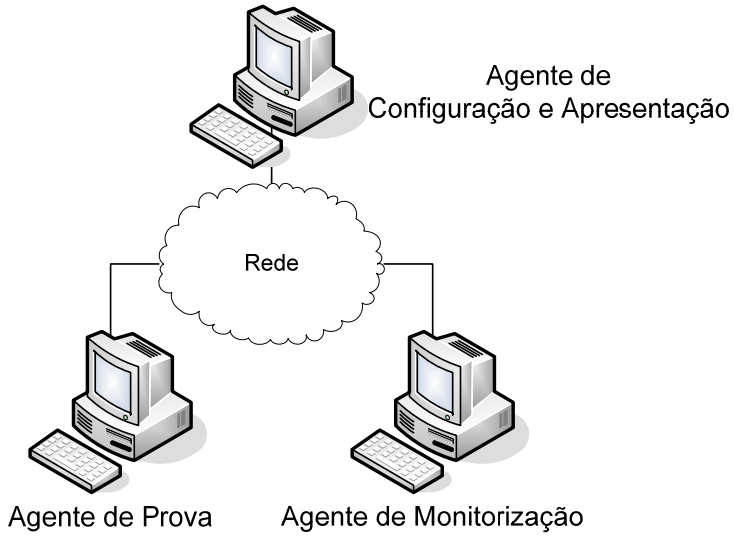


Figura 5-11 Estrutura Lógica de Rede 1

Numa fase seguinte, construiu-se o esquema completo, isto é, implementado também o *Agente de Configuração e Apresentação*. Neste agente foram instaladas as seguintes aplicações:

- Windows XP – SP2 [Version 5.1.2600]
- XAMPP 1.6.4
- ActivePerl 5.8.8.822

Depois da construção deste agente e da sua integração no ambiente, foram posicionados no sistema, vários *Agentes de Prova* interligados através da mesma rede física, conforme é apresentado na Figura 5-12.

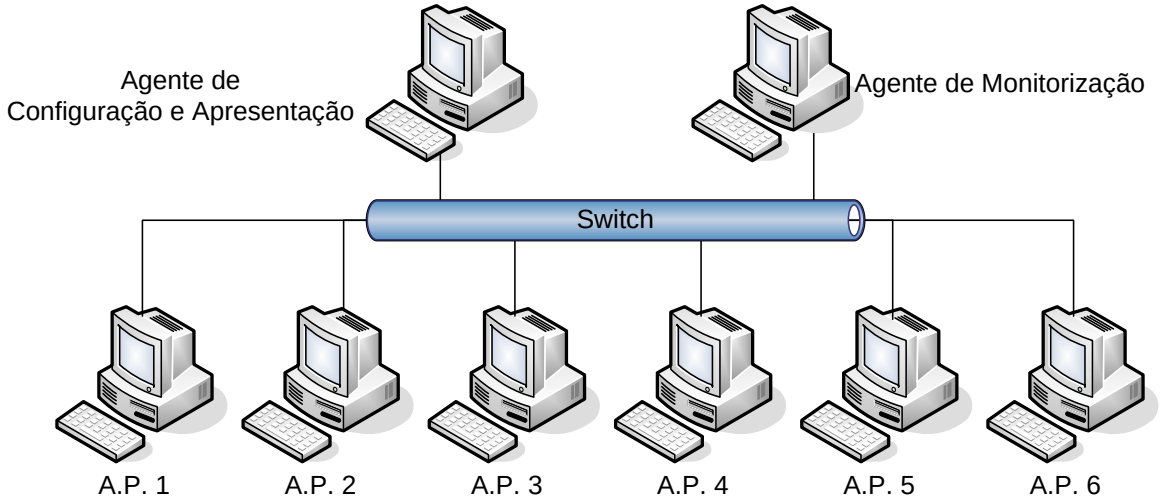


Figura 5-12 Estrutura Lógica de Rede 2

Por último, foi testada a hipótese de posicionar o *Agente de Configuração e Apresentação* numa rede diferente da rede do *Agente de Monitorização* e do *Agente de Prova*, teste apresentado na Figura 5-13. Desta forma, pôde-se analisar o comportamento de todos os agentes, e concluir que os parâmetros especificados para as configurações, serviam para o correcto funcionamento do sistema.

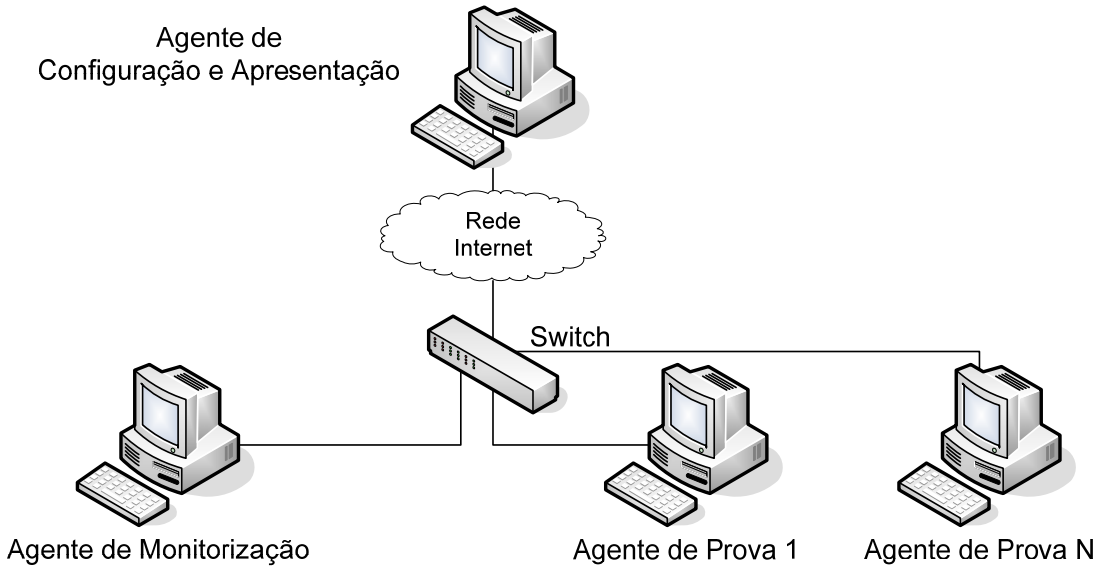


Figura 5-13 Estrutura de Rede Final de Testes

Com este último teste, foi testada a capacidade do sistema ter o *Agente de Configuração e Apresentação* em local remoto. Neste caso, foram realizados testes para verificar exactamente

qual o volume de transferência de informação no caso de um ou vários *Agentes de Prova*, dados que são apresentados no próximo tópico deste documento.

Outros testes foram realizados com mais *Agentes de Prova*, onde se verificou que as exigências de rede eram fáceis de conseguir para um número aceitável de monitorização de ecrãs, ou seja, 9 ecrãs em simultâneo. No tópico seguinte, são apresentados os resultados relevantes do protótipo desenvolvido.

5.4 **- Resultados**

O sistema apresentado sob a forma de protótipo foi submetido a testes em vários cenários e, vários ambientes de rede, que permitiram retirar alguns resultados interessantes. Inicialmente, uma das questões que se tomou como determinante, seria a comunicação entre agentes, mais especificamente, as questões relacionadas com o volume de tráfego e atrasos na comunicação que a solução poderia gerar, tanto na comunicação *Agente de Configuração e Apresentação – Agente de Monitorização*, como na comunicação *Agente de Configuração e Apresentação – Agente de Prova*. Os valores transferência de informação dirigidos ao *Agente de Configuração e Apresentação* tornam-se ainda mais elevados com a presença de mais do que um *Agente de Prova*, o que acaba por submeter ainda maiores exigências à estrutura de rede no que respeita à largura de banda e atrasos. Para avaliar este impacto, foram realizados alguns testes específicos em várias situações. Um dos resultados destes testes é apresentado na Figura 5-14 onde se pode observar um gráfico com o volume de tráfego na interface de rede do *Agente de Configuração e Apresentação*, em três situações específicas. A situação A apresenta o tráfego para zero *Agentes de Prova*, ou seja, não existe tráfego visível na interface. A situação B apresenta o tráfego para um *Agente de Prova* com ligação ao seu ecrã remoto, onde inclusive se pode verificar um pico de 3% do tráfego máximo na interface, que é de 100Mbps, ou seja, chega aos 3Mbps. Em média esta situação apresenta um volume de 0.75% a 1%, ou seja 750Kbps a 1Mbps. Por último, a situação C apresenta o caso de dois *Agentes de Prova* a enviarem os seus fluxos para o *Agente de Configuração e Apresentação*. Neste caso, verifica-se que a média do tráfego observado está em 1.5%, ou seja, 1.5Mbps, onde existem mais picos em 2.5% e 3%, ou seja, 2.5Mbps e 3Mbps.

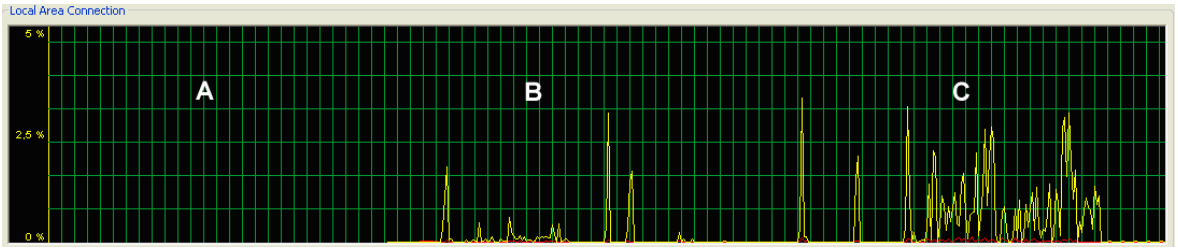


Figura 5-14 Gráfico de Volume de Tráfego – Agente de Configuração e Apresentação e Agente de Prova (eixo vertical - Percentagem de 100Mbps; eixo horizontal – Tempo)

Através dos testes realizados com vários *Agentes de Prova*, conseguiu-se concluir que o sistema apresenta resultados satisfatórios para uma capacidade total de 9 alunos por prova, sem que se deteriore significativamente o seu desempenho. A Figura 5-15 apresenta um exemplo do ecrã do *Agente de Monitorização* onde se podem ver 9 fluxos simultâneos. O caso que implicaria um maior volume de tráfego seria o caso com 10/12 ecrãs, o que inclusive poderia levar a uma necessidade de um ecrã maior, ou até a projecção, visto os ecrãs terem de ser reduzidos no seu tamanho para serem apresentados num só ecrã no *Agente de Configuração e Apresentação*. No caso mais usual, a solução não deveria apresentar mais do que 9 utilizadores em simultâneo, todavia, poderiam ser admitidos mais se as funções do *Agente de Configuração e Apresentação* pudessem ser separadas de forma a existir um Agente de Configuração e vários Agentes de Apresentação (com funções apenas de comunicação de fluxos de ecrãs), que possibilitassem a divisão dos ecrãs monitorizados e do tráfego na rede.

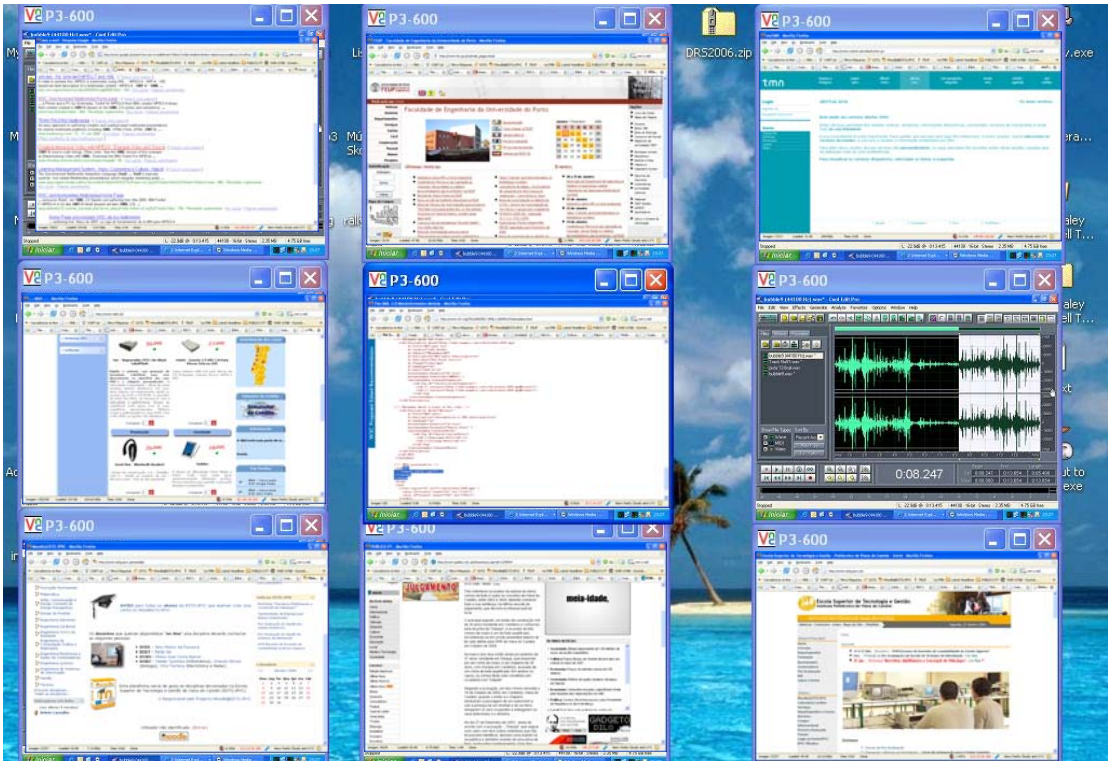


Figura 5-15 Exemplo dos objectos presentes na cena multimédia do protótipo

No caso da comunicação *Agente de Configuração e Apresentação - Agente de Monitorização*, e *Agente de Monitorização – Agente de Prova*, todas as informações que são transportadas são fluxos de caracteres simples e, neste nível, sem qualquer tipo de encriptação. Ou seja, a ligação entre estes agentes não é tão exigente como a ligação do *Agente de Configuração e Apresentação – Agente de Prova*. Assim, conclui-se que estes agentes podem ser instalados em qualquer sítio com acesso à rede de monitorização, sem grande exigência de performance da rede no que respeita a débitos e atrasos na informação.

No próximo capítulo vão ser apresentadas as conclusões essenciais deste trabalho e apresentadas algumas considerações finais que lhe são devidas.

Capítulo 6 - Conclusões e Considerações Finais

Este trabalho surge da necessidade de obtenção de um método automático e expedito para a monitorização electrónica de ambientes de formação, em particular para a fase de avaliação através da realização de provas de exame. A fase de avaliação é uma das fases importantes de uma formação e neste âmbito foram realizadas pesquisas sobre actuais sistemas de prova electrónica. Como resultado da pesquisa, verificou-se que nos sistemas sobre análise as provas são realizadas com recurso a servidores remotos dificilmente configuráveis para algumas especificidades nas variáveis de ambiente a monitorizar. Assim, como suporte para uma dinâmica própria foi formulado um método de monitorização específica, automatizada e dinâmica que permite a definição de critérios antes e durante a prova, bem como actuação em variáveis predefinidas. O modelo é a base fundamental de todo o trabalho apresentado e este tem como princípio a realização de uma monitorização electrónica com incidência em duas variáveis de ambiente específicas consideradas fundamentais: a dinâmica de processos e a imagem dos ecrãs em cada um dos elementos em prova (computador com o examinando). Este modelo foi concebido para que pudesse monitorizar e até actuar em circunstâncias predefinidas relacionadas com estas variáveis de ambiente e estudado para que pudesse incorporar no futuro outras variáveis que possam ser relevantes. Tendo por base a arquitectura do modelo proposto, foi desenvolvido um protótipo que permite a implementação e teste do método enunciado.

Tendo por base o modelo de monitorização proposto, foi realizada uma análise de requisitos e criada uma especificação funcional que perspectiva a implementação de um protótipo que, de forma fiel, consiga traduzir o funcionamento do modelo referido. Aquando a fase de início da realização deste protótipo, surgiram várias arquitecturas possíveis. Estas, foram alvo de várias discussões de trabalho e que levaram ao sistema final, através de um processo iterativo no sentido de aproximar o funcionamento deste sistema ao modelo específico apresentado.

Na arquitectura final, foram implementados os 3 Agentes principais que servem a arquitectura: o *Agente de Configuração e Apresentação*, o *Agente de Monitorização* e o *Agente de Prova* do sistema com recurso a várias tecnologias de suporte. Neste contexto, é de referir que nem todas as directrizes da fase de especificação foram implementadas em cada um destes Agentes no protótipo final. Por outro lado, existiram alterações efectuadas no sistema que não foram previstas na fase de especificação.

Um dos casos, surgiu na fase inicial de desenvolvimento do protótipo, com a ideia de alteração do código do VNC que pareceu complexa mas ao mesmo tempo incontornável para a execução do protótipo. A investigação em torno da aplicação VNC mostrou ser importante para a compreender melhor e verificar as suas potencialidades no campo da monitorização. Além disso, esta investigação permitiu concluir que a interligação a esta aplicação podia realizada através de um ficheiro de arranque simples descrito atrás neste documento e assim, evitar a alteração directa no código do VNC em Java. No entanto, surgiram outros obstáculos na utilização desta aplicação como é o caso da colocação dos ecrãs em posição e terminação de janelas no *Agente de Configuração e Apresentação*. Como consequência disto, no protótipo final, quando os ecrãs são despoletados, ficam todos numa posição central e sobrepostos, necessitando o examinador de os deslocar para conseguir visualizá-los. Da mesma forma, quando se dá ordem para terminar o fluxo de ecrã, o utilizador tem de fechar a janela do ecrã correspondente. Apesar destas duas características serem importantes para um sistema final, do ponto de vista conceptual a sua execução não foi considerada determinante uma vez que não interferem com a dinâmica principal do sistema. Ao mesmo tempo, do ponto de vista do desenvolvimento apresentam-se como funcionalidades que exigiriam profundas alterações a nível gráfico no código VNC do *Agente de Monitorização.* Assim, esta questão pode ser endereçada como trabalho futuro, para um sistema final sem afectar a dinâmica base do protótipo referido.

Durante o processo de desenvolvimento, uma das etapas fundamentais foi a constituição de um elemento central em toda a dinâmica projectada: a linguagem de descrição de cena e ambiente. Esta linguagem teria de permitir de forma directa apresentar todos os elementos do sistema e o modo como estes interagem no esquema de monitorização. Além disso, teria de ser de construção dinâmica, uma vez que o método de monitorização está assente em variáveis de

ambiente que podem modificar a construção da cena em cada instante. Desta forma, foi levado a cabo um estudo sobre linguagens que pudessem descrever da melhor forma o método e os esquemas de monitorização pretendidos. Do estudo realizado, a linguagem SMIL foi apontada como a linguagem que melhor se adaptava aos esquemas de monitorização previstos. Depois desta constatação, foram formulados vários esquemas possíveis de descrição de cena e ambiente e para a sua formação consistente foram utilizados esquemas de validação de gramática estrutural.

Outra componente fundamental que permitiu o desenvolvimento do sistema foi a ferramenta de gestão WMI para a interacção nos processos dos computadores dos *Agentes de Prova*. Esta ferramenta demonstrou ser versátil e directa entre sistemas Microsoft, no entanto, ressalve-se o caso de ser necessária interoperabilidade entre diferentes sistemas. Nesta situação o protocolo SNMP poderia ser adoptado como uma boa alternativa.

Depois de implementado o protótipo procedeu-se à realização de vários testes a funcionalidades e desempenhos, de onde se retiraram resultados interessantes. Dos resultados que foram já descritos, salienta-se a execução de vários *Agentes de Prova* e a sua consequente utilização da rede que levaram a conclusões objectivas em relação ao número de máquinas que se podem colocar em funcionamento no sistema. Num contexto futuro, as exigências de largura de banda e atraso poderiam ser reduzidas caso fosse implementada compressão nas comunicações sobre o protocolo RFB em VNC. Da mesma forma, todo o sistema de fluxos deveria ser processado sobre comunicação segura para que a informação não pudesse ser interceptada, modificada ou utilizada por intermediários não autorizados. Da mesma forma, de futuro também pode ser prevista a separação das funções do *Agente de Configuração e Apresentação* para que constituam dois Agentes separados (*O Agente de Configuração* e o *Agente de Apresentação*). Uma vez que o *Agente de Configuração e Apresentação* implementa as suas funções em blocos distintos, o protótipo desenvolvido está preparado para esta separação e assim, será possível a coexistência de vários Agentes de Apresentação, úteis para melhor monitorizar os ecrãs de VNC.

Em suma, o protótipo construído permitiu que, com as condições que foram sendo criadas ao longo do tempo da sua realização, fosse testado e reformulado em vários ambientes reais numa

evolução contínua. Com os testes realizados foram obtidos resultados interessantes e satisfatórios que remetem às particularidades do método de monitorização formulado.

Como protótipo que é apresentado, o sistema de monitorização construído teria de sofrer alguns melhoramentos para se apresentar como uma aplicação final de monitorização em ambientes de formação com acesso a avaliação por prova electrónica. Em todo o caso, este permitiu implementar, testar e avaliar a implementação do método base especificado e assim, permite uma implementação imediata do conceito de monitorização electrónica patente neste método. Ao mesmo tempo, este trabalho aponta alguns problemas e algumas soluções, possibilidade de melhorias e desenvolvimentos futuros num domínio abrangente como é o da formação electrónica.

Capítulo 7 - Referências

1. Ribie - Rede Iberoamericana de Informática Educativa. *Enquadramento Conceptual de Ensino / Aprendizagem para o Desenvolvimento de Programas Educativos*. 2005 [cited 10-03-2005]; Available from: http://lsm.dei.uc.pt/ribie/docfiles/txt200341733051ENQUADRAMENTO%20CONCEPTUAL.pdf.
2. IEEE. *Institute of Electrical and Electronics Engineers, Inc.* [cited 10-01-2005]; Available from: http://www.ieee.org.
3. ISO. *International Standards Organization (ISO)*. [cited 10-01-2005]; Available from: http://www.iso.org.
4. ADL. *Advanced Distributed Learning*. [cited 10-02-2005]; Available from: http://www.adlnet.gov.
5. ADL. *SCORM - Shareable Courseware Object Reference Model*. [cited 20-03-2005]; Available from: http://www.adlnet.gov/scorm/index.cfm.
6. E-Learning Site. *E-Learning Definition*. [cited 01-02-2005]; Available from: http://www.e-learningsite.com/.
7. Harvi Singh and C. Reed, *A White Paper: Achieving Success with Blended Learning*. 2001.
8. University of Washington. *Tools and Methods for Distance Education*. 2000 [cited 02-05-2005]; Available from: http://www.cs.washington.edu/education/dl/tools/.
9. Adobe. *Macromedia Breeze*. 2006 [cited 21-09-2005]; Available from: http://www.adobe.com/products/breeze/.
10. Moodle. *Moodle Plataform*. 2006 [cited 02-05-2005]; Available from: http://www.moodle.org.
11. Stephen Downes. *E-learning 2.0*. [cited 10-10-2005]; Available from: http://elearnmag.org/subpage.cfm?section=articles&article=29-1.
12. Graduate Management Admission Council. *GMAT - Graduate Management Admission Test*. [cited 12-05-2006]; Available from: http://www.gmac.com/gmac/thegmat/.
13. Pearson. *VUE - Virtual University Enterprises*. [cited 22-04-2006]; Available from: http://www.vue.com/.
14. Thompson. *PROMETRIC*. [cited 22-04-2006]; Available from: http://www.prometric.com.
15. Cisco Systems Inc. *Cisco Systems*. [cited 10-02-2006]; Available from: http://www.cisco.com/.
16. ProcessLibrary. *Libraria de processos*. 2003 [cited 02-05-2005]; Available from: http://www.processlibrary.com/.
17. *Mozilla*. 2006 [cited 01-05-2005]; Available from: http://www.mozilla.com.
18. *Microsoft Corporation*. 2006 [cited 02-10-2005]; Available from: http://www.microsoft.com.
19. Adobe. *Adobe PDF*. 2006 [cited 21-09-2005]; Available from: http://createpdf.adobe.com/.

20. Open-Source-Iniciative. *Open Source Definition*. [cited 10-10-2006]; Available from: http://www.opensource.org/docs/definition_plain.php.
21. Perl. *Perl.com The Source for PERL*. 2006 [cited 02-05-2005]; Available from: http://www.perl.com.
22. PHP. *PHP: Hypertext Preprocessor*. 11-12-2005 [cited 21-09-2005]; Available from: http://www.php.net/.
23. W3C. *Simple Object Access Protocol (SOAP)*. [cited 02-02-2005]; Available from: http://www.w3.org/TR/soap/.
24. Apache Friends. *XAMPP*. 2007 [cited 11-05-2006]; Available from: http://www.apachefriends.org/en/xampp.html.
25. RealVNC. *Fundamentos VNC - Virtual Network Computing*. 2002 [cited 02-05-2005]; Available from: http://www.realvnc.com.
26. Symantec. *PCAnywhere*. [cited 20-04-2006]; Available from: http://www.pcanywhere.com/smb/products/overview.jsp?pcid=cli_mgmt&pvid=pca12.
27. Citrix. *GoToMyPC*. [cited 20-04-2006]; Available from: http://www.gotomypc.com.
28. GNU. *General Public Licence*. 1984 [cited 21-09-2005]; Available from: http://www.gnu.org/copyleft/gpl.html.
29. Tristan Richardson, *RFB - Remote Frame Buffer*. 2005, RealVNC Ltd,.
30. W3C. *World Wide Web Consortium - Extensible Markup Language (XML)*. 1999 [cited 02-05-2005]; Available from: http://www.w3.org/XML/.
31. W3C. *W3C World Wide Web Consortium*. [cited 01-03-2005]; Available from: http://www.w3.org/.
32. MPEG. *MPEG-4, eXtensible MPEG-4 Textual Format - XMT-A*. 2004 [cited 17-11-2005]; Available from: http://www.digitalpreservation.gov/formats/fdd/fdd000156.shtml.
33. MPEG. *MPEG-4, eXtensible MPEG-4 Textual Format - XMT-O*. 2004 [cited 17-11-2005]; Available from: http://www.digitalpreservation.gov/formats/fdd/fdd000156.shtml.
34. W3C. *SMIL - Synchronized Multimedia Integration Language*. 1999 [cited 01-11-2005]; Available from: http://doc.async.com.br/w3-recs/REC-smil/introduction.html.
35. W3C. *Synchronized Multimedia Integration Language (SMIL 2.0)*. 2005 12-11-2005 [cited 06-12-2005]; Available from: http://www.w3.org/TR/2005/REC-SMIL2-20050107/.
36. MPEG. *Moving Picture Experts Group*. 2000 [cited 04-10-2005]; Available from: http://www.mpeg.org/MPEG/index.html.
37. MPEG. *BIFS - BInary Format for Scenes*. 2004 [cited 17-11-2005]; Available from: http://www.chiariglione.org/mpeg/tutorials/papers/icj-mpeg4-si/05-BIFS_paper/5-BIFS_paper.htm.
38. VRML. *VRML - Virtual Reality Modeling Language*. 2000 [cited 11-11-2005]; Available from: http://www.w3.org/MarkUp/VRML/.
39. ISO/IEC. *MPEG-4*. 2002 10-03-2005 [cited 02-05-2005]; Available from: http://www.chiariglione.org/mpeg/standards/mpeg-4/mpeg-4.htm.

40. MPEG. *MPEG-4, eXtensible MPEG-4 Textual Format - XMT*. 2004 [cited 11-10-2005]; Available from: http://www.digitalpreservation.gov/formats/fdd/fdd000156.shtml.

41. Len Bullard, *X3D -Extensible 3D: XML meets VRML*. 2006.

42. Pascal Dayre. *Les formats du multimédia et des hypermédias*. 2003 [cited 12-11-2005]; Available from: http://www.enseeiht.fr/~dayre/hypermedias/Formats.html.

43. Web3d. *X3D*. 2005 21-05-2005 [cited 02-05-2005]; Available from: http://www.web3d.org/x3d/.

44. W3C. *XML Schema*. 2004 04-03-2005 [cited 02-05-2005]; Available from: http://www.w3.org/XML/Schema.

45. ISO. *Schematron*. 2004 02-05-2005 [cited 02-05-2005]; Available from: http://www.schematron.com/.

46. Microsoft. *WMI - Windows Management Instrumentation*. 2006 [cited 19-12-2005]; Available from: http://www.microsoft.com/whdc/system/pnppwr/wmi/default.mspx.

47. DMTF. *Common Information Model*. 2002 [cited 21-09-2005]; Available from: http://www.wbemsolutions.com/tutorials/CIM/cim.html.

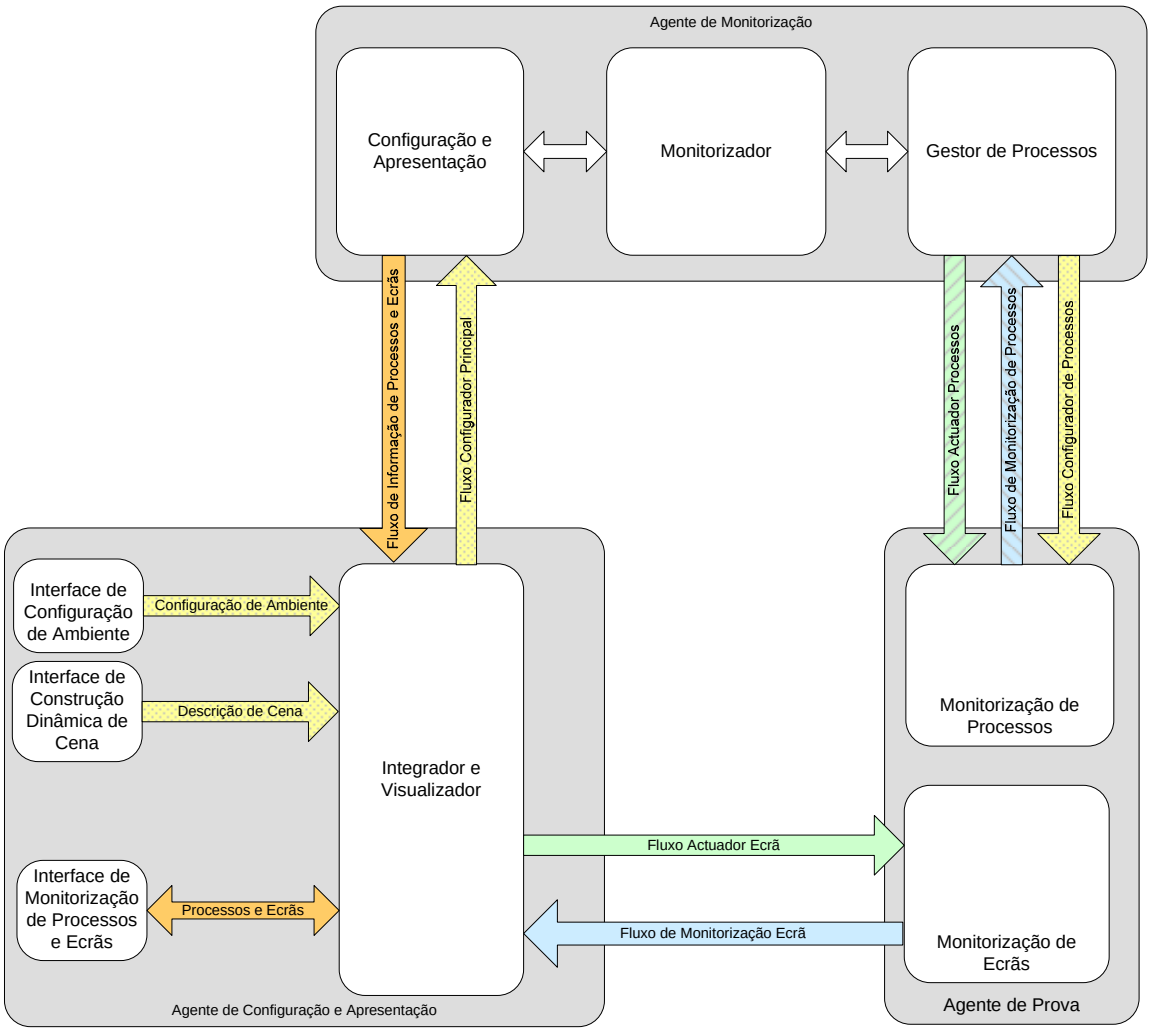
48. SNMP. *Simple Network Management Protocol*. 2005 [cited 2006 02-10-2005]; Available from: RFC 1157 - Simple Network Management Protocol (SNMP).

49. Roth Consulting. *Win 32 Perl Programing*. 2005 [cited 02-05-2005]; Available from: http://www.roth.net/.

50. IEEE, *Ethernet Standard 802.3*. 2002, IEEE.

Capítulo 8 - Anexos

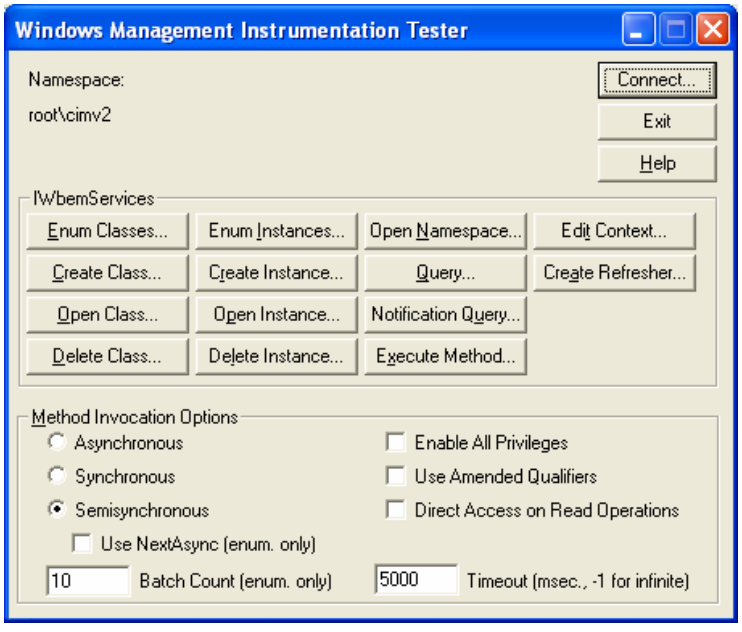
8.1 - Diagrama Geral



8.2 – Instalação do Módulo Win32:API (Perl)

```
ppm> install Win32::API 1
Package 1:
====================
Install 'Win32-API' version 0.41 in ActivePerl 5.8.6.811.
====================
Downloaded 36283 bytes.
Extracting 22/22: blib/arch/auto/Win32/API/Callback/Callback.lib
Installing C:\Perl\site\lib\auto\Win32\API\API.bs
Installing C:\Perl\site\lib\auto\Win32\API\API.dll
Installing C:\Perl\site\lib\auto\Win32\API\API.exp
Installing C:\Perl\site\lib\auto\Win32\API\API.lib
Installing C:\Perl\site\lib\auto\Win32\API\Callback\Callback.bs
Installing C:\Perl\site\lib\auto\Win32\API\Callback\Callback.dll
Installing C:\Perl\site\lib\auto\Win32\API\Callback\Callback.exp
Installing C:\Perl\site\lib\auto\Win32\API\Callback\Callback.lib
Installing C:\Perl\html\site\lib\Win32\API.html
Installing C:\Perl\html\site\lib\Win32\API\Callback.html
Installing C:\Perl\html\site\lib\Win32\API\Struct.html
Installing C:\Perl\html\site\lib\Win32\API\Type.html
Files found in blib\arch: installing files in blib\lib into architecture depende
nt library tree
Installing C:\Perl\site\lib\Win32\API.pm
Installing C:\Perl\site\lib\Win32\API\Callback.pm
Installing C:\Perl\site\lib\Win32\API\Struct.pm
Installing C:\Perl\site\lib\Win32\API\Type.pm
Successfully installed Win32-API version 0.41 in ActivePerl 5.8.6.811.
ppm>
```

8.3 – Software de Teste WMI - wmimgmt.msc



8.4 **- Código do Agente de Monitorização (Perl)**

```
# use

# MEPE-INTER-v1.pl -scene=scenedesc_v2_smil.xml -output=output.xml -opt=x

$/="\n";
$,=" ";

print "Script[0] - Initializing\n";

use XML::Simple;
use Getopt::Long;

my $scenefile = '';
my $output = '';
my $opt= '';

print "Script[1] - Catching Options";

GetOptions ('scene=s' =>\$scenefile, 'output=s' => \$output, 'opt=s'=> \$opt);

print " - OK\n";

#ficheiro ou ecran?
if($output=~/\w /){
    $out=$output;
    open(Fout, ">$out") or die "Error Creating or Writing to File: $file";
    print "OPTIONS -> OUTPUT=$out";
    select Fout;}
else{
    $out='STDOUT';
    print "OPTIONS ->Output=$out";}

print "\t->SceneFile=$scenefile\t->Option=$opt\n";
print " - > Script Progress 3 - Capturing Scene Description: $scenefile\n";


#ler ficheiro de cena
my $scene = XMLin("$scenefile", forcearray =>['body']);

#open(F_HOSTS, "<$hostsfile") or die "File Open Error - File: $hostsfile";
#while (<F_HOSTS>){print "$_";}
#print "\n";

$eventos=0;
print " - > Script Progress 4 - Getting XML Values\n";
while (my($chave1, $valor1) = each(%$scene)) {
    print (" 1 Chave: $chave1\t\tValor: $valor1\n");
    if ($chave1 eq "body") {
       $body = $valor1;
       foreach $tseq (@$body){
          while(my($chave2, $valor2) = each (%$tseq)){
             print (" 2 Chave: $chave2\t\tValor: $valor2\n");
             while(my($chave3, $valor3) = each (%$valor2)){
                print (" 3 Chave: $chave3\t\tValor: $valor3\n");
                if ($chave3 eq "repeatCount"){
                        if($valor3 eq "indefinite"){$repeat=0;}
                        else {$repeat=$valor3;}
                }
                if ($chave3 eq "img") {

                        foreach $imgitem (@$valor3) {
                                $eventos++;
                                $computador[$eventos]=$imgitem->{computador};
                                $janela[$eventos]=$imgitem->{janela};
                                $begin[$eventos]=$imgitem->{begin};
                                $dur[$eventos]=$imgitem->{dur};

                                print "$eventos -
comp[$computador[$eventos]]\tjanela[$janela[$eventos]]\tbegin[$begin[$eventos]]\tduring[$dur[$eve
ntos]]\n";

                        }
                }
             }
          }
       }
    }
}
```

```
print "Eventos=$eventos\n";
$scene_time=0;

for ($j==1; $j<=$eventos; $j++) {
   #print "----- $begin[$j]+$dur[$j]\n";
   if (($begin[$j]+$dur[$j])>$scene_time) {
      $scene_time=($begin[$j]+$dur[$j]);
   }
}
print "Scene_time=$scene_time\trepeat=$repeat\n";

if ($opt==3) {
   if ($repeat==0) {$total_time=-1;}
   else {$total_time=$scene_time*$repeat;}

   for ($clock=0 ; ($clock<=$total_time || $total_time==-1)  ; $clock++) {
   print "\tCLOCK=$clock seg\n";
      $j=0;

      for ($mark=1;$mark<=$eventos ;$mark++) {
         $multiplo=$clock%$scene_time;

         if ($begin[$mark]==$clock || $begin[$mark]==$multiplo) {
         print "Start\tcomp[$computador[$mark]] on window[$janela[$mark]] - begin
at=$begin[$mark]\n";
         }
         if ($begin[$mark]+$dur[$mark]==$clock || $begin[$mark]+$dur[$mark]==$multiplo ||
($begin[$mark]+$dur[$mark])%$scene_time==$multiplo) {
         print "Stop\tcomp[$computador[$mark]] on window[$janela[$mark]] - close
at=$begin[$mark]+$dur[$mark]\n";
         }

      }

#  sleep 1;
   }
}
```

8.5 **- Código do Agente de Configuração e Apresentação (PHP)**

```
<html><head>
<title>DescCena - InterfCena</title>
</head>
<body>
<b>Introdução de Descrição de Ambiente + Cena - MEPE</b><br><br>
<?php
$moncol = $_POST["moncol"];
$monlin = $_POST["monlin"];

$nome1= $_POST["nome1"];
$ip1  = $_POST["ip1"];
$nome2= $_POST["nome2"];
$ip2  = $_POST["ip2"];
$nome3= $_POST["nome3"];
$ip3  = $_POST["ip3"];

$proc1  = $_POST["proc1"];
$proc2  = $_POST["proc2"];
$nproc1  = $_POST["nproc1"];
$nproc2  = $_POST["nproc2"];

$dur  = $_POST["dur"];
$evi  = $_POST["evi"];

if ( $moncol == "" || $monlin == "" || $nome1 == "" || $ip1 == "") echo "INFO - Dados em falta -
Erro no preenchimento";
else
{echo "INFO - Dados inseridos" ;

$myFile = "SceneFile.smil";
$fh = fopen($myFile, 'w') or die("can't open file");

$enter="\n";
fwrite($fh, "<smil>\n<head>\n<monitor linhas=\"$monlin\" colunas=\"$moncol\"/>\n");
fwrite($fh, $enter);
$jan="0";
$lin="0";
while($lin != $monlin)
{   $col="0";
```

```
   $lin++;
   while ($col != $moncol)
   {  $jan++;
      $col++;
      fwrite($fh, "<src janela=\"$jan\" linha=\"$lin\" coluna=\"$col\"/>\n");}
}

fwrite($fh, $enter);

if($nome1 != "" || $ip1 != "") fwrite($fh, "<src computador=\"1\" formando=\"$nome1\"
ip=\"$ip1\"/>\n");
if($nome2 != "" || $ip2 != "") fwrite($fh, "<src computador=\"2\" formando=\"$nome2\"
ip=\"$ip2\"/>\n");
if($nome3 != "" || $ip3 != "") fwrite($fh, "<src computador=\"3\" formando=\"$nome3\"
ip=\"$ip3\"/>\n");
fwrite($fh, $enter);

fwrite($fh, "<meta var=\"proc_perm\"/>\n");
if($proc1 != "") fwrite($fh, "<processo> $proc1</processo>\n");
if($proc2 != "") fwrite($fh, "<processo> $proc2</processo>\n");

fwrite($fh, "<meta var=\"proc_nperm\"/>\n");
if($nproc1 != "") fwrite($fh, "<processo> $nproc1</processo>\n");
if($nproc2 != "") fwrite($fh, "<processo> $nproc2</processo>\n");
fwrite($fh, $enter);

fwrite($fh, "</head>\n<body>\n<seq repeatCount=\"indefinite\">\n");

$j="0";
$beg="0";
if($nome1 != "" || $ip1 != ""){
   while($j != $jan)  {
      $j++;
      fwrite($fh, "<video computador=\"1\" janela\"$j\" begin=\"$beg\" dur=\"$dur\"/>\n");
      $beg=$beg+$dur;
      }
}

$j="0";
$beg="0";
if($nome2 != "" || $ip2 != ""){
   while($j != $jan)  {
      $j++;
      fwrite($fh, "<video computador=\"2\" janela\"$j\" begin=\"$beg\" dur=\"$dur\"/>\n");
      $beg=$beg+$dur;
      }
}

$j="0";
$beg="0";
if($nome3 != "" || $ip3 != ""){
   while($j != $jan)  {
      $j++;
      fwrite($fh, "<video computador=\"3\" janela\"$j\" begin=\"$beg\" dur=\"$dur\"/>\n");
      $beg=$beg+$dur;
      }
}
fwrite($fh, "</seq>\n</body>\n</smil>\n");
fclose($fh);
}
?>

<br><br>
<form action="cena.php" method="post">

<b>Monitor:</b><br>
Linhas: <input type="text" size=4 name="monlin"/>  Colunas: <input type="text" size=4
name="moncol"/> <br><br>
<b>Computadores:</b><br>
Comp1 - Nome: <input type="text" size=30 name="nome1"/>  IP: <input type="text" size=15
name="ip1"/> <br>
Comp2 - Nome: <input type="text" size=30 name="nome2"/>  IP: <input type="text" size=15
name="ip2"/> <br>
Comp3 - Nome: <input type="text" size=30 name="nome3"/>  IP: <input type="text" size=15
name="ip3"/> <br>
<br>
<b>Processos Permitidos:</b><br>
Proc1: <input type="text" size=30 name="proc1"/> <br>
Proc2: <input type="text" size=30 name="proc2"/> <br><br>
<b>Processos Não Permitidos:</b><br>
Proc1: <input type="text" size=30 name="nproc1"/> <br>
Proc2: <input type="text" size=30 name="nproc2"/> <br><br>
```

```
<b>Cena:</b><br>
Duração de cada elemento: <input type="text" size=5 name="dur"/>
Elemento em evidência: <input type="text" size=5 name="evi"/> <br>
<br><br>
<input type="submit" value="Enviar Descrição de Ambiente + Cena" />
</form>
</body>
</html>
```

8.6 **- Código de Descrição de Ambiente e Cena**

```
<smil><head>
<monitor linhas="3" colunas="4"/>

<src janela="1" linha="1" coluna="1"/>
<src janela="2" linha="1" coluna="2"/>
<src janela="3" linha="1" coluna="3"/>
<src janela="4" linha="1" coluna="4"/>
<src janela="5" linha="2" coluna="1"/>
<src janela="6" linha="2" coluna="2"/>
<src janela="7" linha="2" coluna="3"/>
<src janela="8" linha="2" coluna="4"/>
<src janela="9" linha="3" coluna="1"/>
<src janela="10" linha="3" coluna="2"/>
<src janela="11" linha="3" coluna="3"/>
<src janela="12" linha="3" coluna="4"/>

<src computador="1" formando="Pedro Pinto" ip="192.168.1.2"/>
<src computador="2" formando="Antonio Coelho" ip="192.168.1.3"/>
<src computador="3" formando="André Silva" ip="192.168.1.4"/>

<meta var="proc_perm"/>
<processo> explorer.exe</processo>
<processo> calc.exe</processo>
<meta var="proc_nperm"/>
<processo> messenger.exe</processo>
<processo> msn.exe</processo>

</head>
<body>
<seq repeatCount="indefinite">
<video computador="1" janela"1" begin="0" dur="3"/>
<video computador="1" janela"2" begin="3" dur="3"/>
<video computador="1" janela"3" begin="6" dur="3"/>
<video computador="1" janela"4" begin="9" dur="3"/>
<video computador="1" janela"5" begin="12" dur="3"/>
<video computador="1" janela"6" begin="15" dur="3"/>
<video computador="1" janela"7" begin="18" dur="3"/>
<video computador="1" janela"8" begin="21" dur="3"/>
<video computador="1" janela"9" begin="24" dur="3"/>
<video computador="1" janela"10" begin="27" dur="3"/>
<video computador="1" janela"11" begin="30" dur="3"/>
<video computador="1" janela"12" begin="33" dur="3"/>
<video computador="2" janela"1" begin="0" dur="3"/>
<video computador="2" janela"2" begin="3" dur="3"/>
<video computador="2" janela"3" begin="6" dur="3"/>
<video computador="2" janela"4" begin="9" dur="3"/>
<video computador="2" janela"5" begin="12" dur="3"/>
<video computador="2" janela"6" begin="15" dur="3"/>
<video computador="2" janela"7" begin="18" dur="3"/>
<video computador="2" janela"8" begin="21" dur="3"/>
<video computador="2" janela"9" begin="24" dur="3"/>
<video computador="2" janela"10" begin="27" dur="3"/>
<video computador="2" janela"11" begin="30" dur="3"/>
<video computador="2" janela"12" begin="33" dur="3"/>
<video computador="3" janela"1" begin="0" dur="3"/>
<video computador="3" janela"2" begin="3" dur="3"/>
<video computador="3" janela"3" begin="6" dur="3"/>
<video computador="3" janela"4" begin="9" dur="3"/>
<video computador="3" janela"5" begin="12" dur="3"/>
<video computador="3" janela"6" begin="15" dur="3"/>
<video computador="3" janela"7" begin="18" dur="3"/>
<video computador="3" janela"8" begin="21" dur="3"/>
<video computador="3" janela"9" begin="24" dur="3"/>
<video computador="3" janela"10" begin="27" dur="3"/>
<video computador="3" janela"11" begin="30" dur="3"/>
<video computador="3" janela"12" begin="33" dur="3"/>
</seq>
</body></smil>
```