Prototipagem rápida de ferramentas de autoria e visualização para Computação Gráfica

Carlos Afonso Rebelo[1] António Fernando Coelho[1] Filipe Cruz[1]

António Augusto de Sousa[1,2] Fernando Nunes Ferreira[1]

[1]Faculdade de Engenharia da Universidade do Porto, [2]INESC Porto
Rua Dr. Roberto Frias, Porto
{ei99056,acoelho,fcruz,aas,fnf}@fe.up.pt

Sumário

A modelação de ambientes urbanos virtuais é uma necessidade cada vez mais presente no grande público. Exemplos como o Google Earth mostram o seu potencial num mercado de massas, até agora desconhecedor deste tipo de tecnologias. A modelação expedita é uma das soluções para a redução dos custos e dos prazos de execução em projectos deste género. Em modeladores expeditos, baseados em sistemas de produções, como é o caso do modelador XL3D, a especificação dos processos de modelação possui uma complexidade elevada. Este facto motiva o desenvolvimento de ferramentas de autoria que assistam a tarefa.

Neste artigo é apresentado um método para prototipagem rápida de ferramentas gráficas de autoria de especificações de modelação, exemplificado pela sua aplicação ao modelador XL3D. Através da combinação de metodologias de desenvolvimento ágeis com frameworks foi produzida uma ferramenta de autoria, denominada XML Navi, que permite diminuir drasticamente a complexidade da especificação dos processos de modelação. Esta ferramenta possibilita um elevado incremento de produtividade, sobretudo ao nível da definição dos sistemas L geoespaciais que são utilizados para os processos de modelação, pois automatiza a sua tradução da notação nativa para o schema XL3D. O método desenvolvido possui elevada flexibilidade, permitindo a prototipagem de diversos tipos de aplicações de Computação Gráfica, tal como é demonstrado com o desenvolvimento de um protótipo de uma ferramenta de visualização interactiva para ambientes urbanos virtuais.

Palavras-chave

Prototipagem rápida, ferramentas de autoria, modelação expedita, modelador XL3D, visualização interactiva.

1. INTRODUÇÃO

A modelação expedita de ambientes urbanos virtuais visa solucionar os problemas relacionados com a elevada intervenção de recursos humanos qualificados, que é usualmente necessária na modelação tridimensional deste tipo de cenas.

Em sistemas de modelação expedita, como o modelador XL3D, baseado em sistemas L geoespaciais [Coelho06a], a especificação do processo de modelação assenta unicamente num documento, neste caso específico, baseado em XML.

Este tipo de arquitectura, altamente interoperável, requer que todo o conhecimento do utilizador tenha que ser codificado em XML, o que se torna impraticável de realizar manualmente para uma utilização minimamente produtiva. Por outro lado, a própria definição dos sistemas L geoespaciais, que servem de base ao processo de modelação expedito, é complexa, requerendo ferramentas de autoria que auxiliem visualmente o utilizador.

Esta problemática agrava-se durante a fase de desenvolvimento deste tipo de modeladores, em que o próprio modelador, enquanto não atinge um estado de maturação mais avançado, requer a criação e realização de testes. Para a realização destes testes, ou se criam especificações de modelação manualmente, ou então será necessário o desenvolvimento de ferramentas de autoria.

Paralelamente, a procura de soluções de visualização de ambientes urbanos virtuais nem sempre é satisfeita com os visualizadores existentes, sendo a ausência de capacidades de interacção a partir do exterior um dos principais entraves, levando ao desenvolvimento de aplicações específicas.

Felizmente, a crescente procura de soluções informáticas por parte das pequenas organizações, bem como a necessidade de reduzir os custos e de diminuir os tempos envolvidos na transferência de tecnologias inovadoras para produtos, tem potenciado o aparecimento de metodologias mais rápidas e simples para a produção de *software*. Nesta abordagem ao desenvolvimento de *software*, materializada em metodologias como Extreme Programming (XP) ou Rapid Application Development (RAD), a

prototipagem rápida é uma das técnicas que se destaca [Boehm03].

No presente documento apresenta-se o processo de prototipagem rápida de ferramentas gráficas para autoria de especificações XL3D e visualização interactiva de ambientes urbanos virtuais em X3D [X3D].

Na secção seguinte são avaliadas ferramentas de autoria XML e de visualização X3D que têm vindo a ser desenvolvidas para estes fins, assim como as tecnologias que estão na base deste trabalho: o modelador XL3D e a RCP Eclipse [RCPEclipse].

O XML Navi, apresentado na secção 3, é uma aplicação desenvolvida para permitir uma criação e edição mais simples e eficaz das especificações de modelação XL3D. São várias as vantagens que disponibiliza, mas a que mais se destaca é a conversão automática de regras de produção, da notação nativa para XML.

Aplicando os conhecimentos adquiridos durante o desenvolvimento da primeira ferramenta, foi desenvolvida uma outra aplicação gráfica, que permite a visualização dos modelos X3D gerados, e que se apresenta na secção 4. Esta aplicação incorpora o *browser* Java Xj3D [Xj3D] e, através da *Scene Access Interface* (SAI), uma API definida na especificação do X3D permite interacções complexas entre entidades externas e a cena 3D, em tempo real.

2. TRABALHO RELACIONADO

O XML é uma tecnologia ubíqua nos ambientes tecnológicos actuais. Ao nível da Computação Gráfica, por exemplo, o XML foi utilizado como formato base da especificação X3D, evolução natural do VRML. Na área do desenvolvimento é utilizado para *scripts*, ficheiros de configuração, meta-informação e meta-programação, enquanto ao nível da *Web* está na base dos *web-services*, *feeds*, sendo o próprio XHTML baseado em XML. Nos ambientes de produção permite a integração de dados e comunicação entre sistemas heterógeneos, ou seja, é um pilar tecnológico estrutural nos sistemas de hoje.

Os editores visuais de XML distinguem-se dos demais por providenciaremß formulários de acesso à informação construídos "*on-the-fly*" de acordo com o Document Type Definition (DTD) ou XML-Schema Definition (XSD) [XML-Schema] (documentos que definem a estrutura válida do documento XML através da descrição dos elementos e respectivos atributos), evitando desta forma que os utilizadores tenham que lidar directamente com a sua estrutura interna. Estas ferramentas apresentam algum grau de personalização permitindo, por exemplo, a criação de interfaces específicas para determinados *schemas*.

Exemplos deste tipo de ferramentas são:

- **XAmple** – criada por Felix Golubov utiliza o Xerces2 Java Parser da Apache [XAmple];
- **Xeena** – criada pela IBM usa o Xerces2 Java Parser da Apache. É o editor usado pelo Web 3D Consortium para editar ficheiros Extensible 3D (X3D) [Xeena];
- **XMLSpy** – alternativa comercial criada pela empresa Altova e reconhecida como um standdart da indústria, dada a gama de soluções integradas que oferece [XMLSpy].

Curiosamente, dado que o XML é essencialmente uma tecnologia de "bastidor" utilizada ao nível dos programadores e da própria comunicação automática entre sistemas informáticos, o campo dos editores visuais genéricos não tem assistido a grandes evoluções. A título de exemplo, o editor Xeena da IBM, um dos mais promissores, remonta ao ano 2000 e assenta sobre Swing. Ultimamente, o que se verifica é que os produtores de *software* apostam na criação de editores visuais específicos assumindo o papel de ferramentas de autoria, ao invés da adaptação destas ferramentas genéricas.

O X3D é a especificação definida pelo consórcio Web3D [Web3D] para a transmissão de modelos 3D através da *Web*. É uma evolução do VRML e tem vindo a suplantá-lo. Um conjunto de *plugins* possibilita a visualização de documentos X3D utilizando um vulgar programa de navegação para a *Web* (*browser*). Há, no entanto, a registar, a falta de uniformização entre os vários produtores deste tipo de *software* relativamente à interpretação da especificação X3D. As várias empresas competem com o objectivo de desenvolver tecnologias que lhes dêem domínio sobre o apetecível mercado da Web 3D e esta falta de cooperação tem contribuído para o facto de ainda não se ter criado uma base tecnologica forte sobre a qual possam ser dados novos passos no sentido de levar estas tecnologias aos mercados de massas [Leavitt06].

A falta de maturidade do *software* de visualização X3D causa problemas ao desenvolvimento de aplicações que utilizem metáforas de interacção mais complexas do que as disponibilizadas por estes produtos. No entanto, têm surgido alguns projectos que tentam colmatar estas lacunas.

Em [Figueroa05] são apresentadas extensões ao X3D para suportar aplicações complexas de realidade virtual, nas quais novas técnicas e dispositivos de interacção possam ser alterados de forma simplificada. São utilizados componentes reutilizáveis e portáveis juntamente com uma *framework*, e parte da inovação reside no facto de serem usados dispositivos de interacção não convencionais, como varinhas para posicionamento 3D (*wand interaction device*) e sistemas de reconhecimento de voz.

Para aplicações mais complexas, a *Scene Structure and Integration Modelling Language* (SSIML) é uma extensão de UML que pode ser integrada em ferramentas como Rational Rose e que facilita o desenvolvimento de aplicações 3D complexas ao estabelecer uma interface comum aos programadores e modeladores 3D [Vitzthum05].

No projecto VLS (Virtual Learning Space) [LeBlanc05] é apresentada a evolução de um campus virtual, da universidade de Nagoya – JEWELS (Japanese Educational World-wide Electronic Learning System), criado com Active Worlds ®, no qual são exploradas várias tecnologias e a sua integração num ambiente *Web*, com particular destaque para os conteúdos multimédia.

2.1 Prototipagem Rápida de aplicações Gráficas – Arquitectura RCP

Após o crescimento exponencial de aplicações totalmente baseadas em ambientes *Web,* a que se assistiu nos últimos anos, algumas limitações da associação entre DHTML e JavaScript, como a falta variedade nos componentes gráficos, tempos de resposta pouco fiáveis e complexidade elevada, levaram ao aparecimento de abordagens *rich client.*

A Eclipse Rich Client Platform (RCP) [EclipseRCP], que está na base do desenvolvimento do padrão industrial Eclipse IDE, foi doada pela IBM à Eclipse Foundation em 2004. Desde esse momento, o seu uso e da respectiva biblioteca de componentes gráficos (SWT) aumentou exponencialmente.

A Eclipse RCP é particularmente vantajosa em ambientes que conciliem várias tecnologias. Dada a sua arquitectura modular permite integrar, sob o mesmo GUI, tecnologias de natureza diferente e, graças à biblioteca gráfica SWT especialmente desenvolvida para a plataforma, a riqueza gráfica é elevada quer a nível estético quer tecnológico.

Em termos de apresentação, a utilização do SWT distingue-se da concorrência em alguns pontos vitais:

- Componentes com implementação nativa, o que resulta num GUI consistente com o sistema operativo da máquina cliente;
- Melhor desempenho dos componentes gráficos, dada a natureza nativa;
- Adição de bibliotecas gráficas específicas baseadas em SWT, por exemplo UIForms, que incrementam a riqueza visual e funcional.

Ao nível do desenvolvimento de ferramentas na área da Computação Gráfica, é possível retirar inúmeras vantagens desta solução relativamente ao seu concorrente directo, a biblioteca Swing, na prototipagem rápida de aplicações com base em metodologias ágeis. A *framework* RCP é totalmente baseada numa arquitectura de *plugins* que incentiva estratégias de desenvolvimento "*loosely coupled*" que, por sua vez, aumentam a expansibilidade das soluções e a reutilização de componentes (Figura 1).

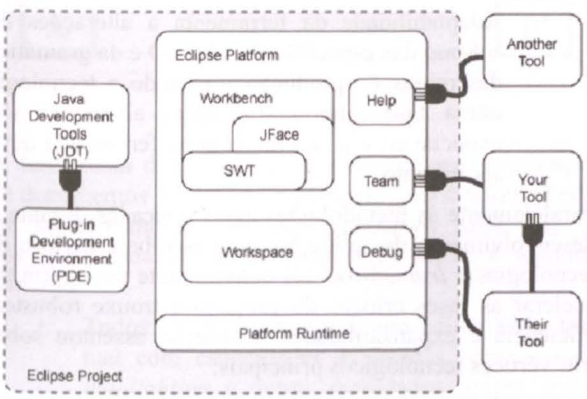

Figura 1 - Arquitectura da Eclipse RCP

Sobre a biblioteca base de componentes gráficos SWT está disponível uma outra camada MVC, denominada JFace, que permite a utilização de componentes gráficos complexos com uma quantidade mínima de código. Por outro lado, através da utilização do Generic Workbench, que constitui a camada de código opcional responsável pela gestão do GUI à luz das regras gráficas do Eclipse IDE, é possível focar as atenções no código realmente importante. A figura 2 detalha a arquitectura geral da plataforma.

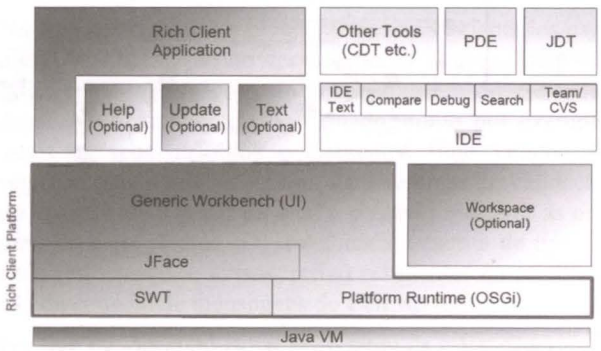

Figura 2 - Arquitectura detalhada da Eclipse RCP

Dado que a aplicação original da qual derivou a plataforma Eclipse IDE se destinava a satisfazer as necessidades de uma comunidade técnica muito exigente (programadores Java especialistas) a SWT contém uma grande variedade de componentes gráficos que se adaptam a uma grande variedade de circunstâncias.

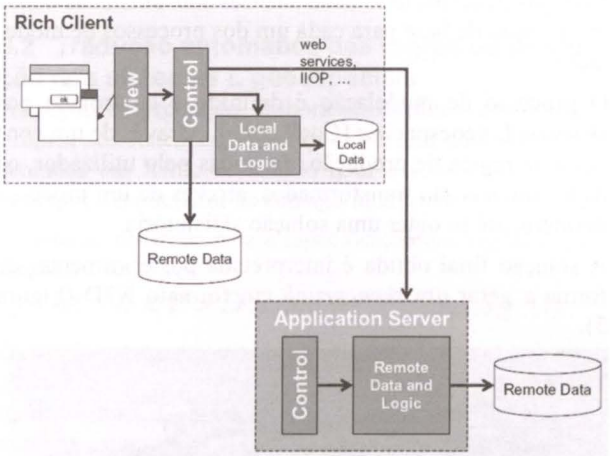

Figura 3 - Exemplo de uma arquitectura Rich Client

Se as aplicações forem construídas seguindo as boas práticas desta plataforma, reutilizando *frameworks*, componentes e código desenvolvidas por terceiros, sempre que possível, desenvolvem-se protótipos muito rapidamente, plenos de funcionalidade e eficiência. A figura 3 apresenta um exemplo de arquitectura que tira partido da RCP Eclipse.

2.2 O modelador XL3D

O modelador XL3D [Coelho04] é um modelador expedito orientado para a geração de ambientes urbanos virtuais, que responde a uma especificação de modelação

gerando de forma automática, um modelo tridimensional em formato X3D (Figura 4).

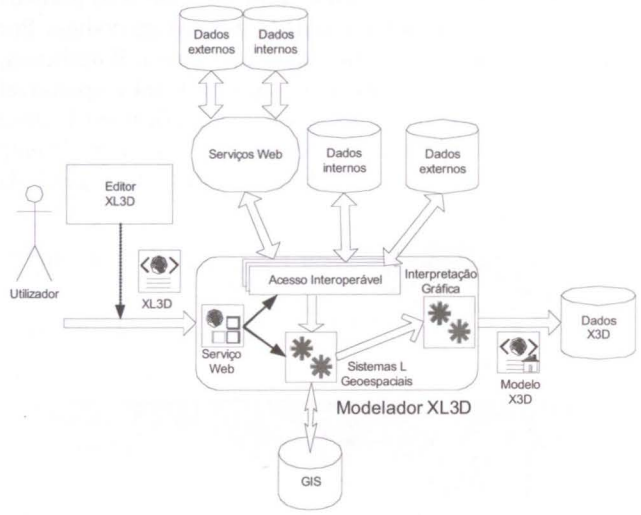

Figura 4 – Arquitectura do modelador XL3D

O sistema baseia-se numa abordagem procedimental, onde toda a modelação é especificada de forma declarativa, através de documentos baseados num XML-Schema denominado XL3D [Coelho03] .

Os dados para modelação são acedidos de forma interoperável, em formatos baseados em XML, e são convertidos, através de transformações XSLT, em cadeias de módulos geoespaciais. Estas cadeias são utilizadas como informação de base para cada um dos processos de modelação.

O processo de modelação é definido e controlado por sistemas L geoespaciais [Coelho06a]. Através de um conjunto de regras de produção, definidas pelo utilizador, os dados iniciais são transformados, através de um processo iterativo, até se obter uma solução satisfatória.

A solução final obtida é interpretada posteriormente, de forma a gerar um *scenegraph* em formato X3D (Figura 5).

Figura 5 – Ambiente urbano virtual gerado pelo modelador XL3D

3. FERRAMENTA DE AUTORIA PARA ESPECIFICAÇÕES DE MODELAÇÃO

Analisando o principal *input* do modelador XL3D e o fluxo de tarefas a este associado, identificaram-se áreas críticas nas quais uma ferramenta de autoria poderia introduzir ganhos consideráveis de tempo e eficácia:

- Edição do ficheiro XML de especificação de modelação de forma contextualizada;
- Escrita das regras de produção em linguagem natural com conversão automática para XML;
- Pré-visualização de resultados de forma a guiar a escrita das regras de produção.

Apesar da difusão da tecnologia XML, os editores disponíveis não apresentam as características necessárias para servirem de base a uma aplicação de autoria para este tipo de modelador. Um editor meramente de texto não seria indicado e os editores visuais genéricos existentes (p.e. Xeena da IBM) não contemplam a integração de casos mais complexos através de novos formulários de edição personalizados.

Partindo destas conclusões foi idealizada uma ferramenta de autoria, denominada XML Navi, que assenta sobre a RCP Eclipse e potencia a utilização da tecnologia XL3D ao resolver as questões indicadas (Figura 6).

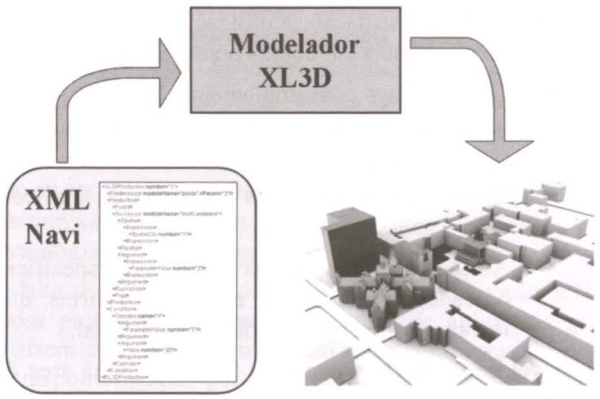

Figura 6 - Enquadramento do XML Navi

Este protótipo foi desenvolvido com dois princípios fundamentais em mente:

- Reutilização de componentes / *frameworks* – única forma de garantir a expansibilidade e a robustez da solução reduzindo o tempo de desenvolvimento;
- Adaptabilidade da ferramenta a alterações do *schema* das especificações XL3D e da gramática das regras de produção – estando a tecnologia numa fase embrionária, estas alterações são expectáveis e o seu impacto na ferramenta deve ser mínimo.

Paralelamente às metodologias ágeis aplicadas durante o desenvolvimento da aplicação, uma escolha criteriosa de tecnologias e *frameworks* foi determinante pois permitiu acelerar as fases críticas do projecto e trouxe robustez, eficiência e expansibilidade. A solução assentou sobre três vértices tecnológicos principais:

- **Eclipse RCP** – *framework* que permite desenvolver aplicações de forma modular e escalável;

- **XMLBeans** – tecnologia de "*data binding*" que permite o mapeamento automático entre estruturas XML e classes Java;
- **JavaCC & JJTree** – Um gerador de *parsers* que permite converter especificações formais de gramáticas em programas Java capazes de construir árvores sintáticas.

3.1 Edição visual da especificação XL3D

Na tecnologia XL3D apresentada, o principal *input* do modelador é um ficheiro XML, denominado especificação de modelação XL3D [Coelho03]. Este ficheiro compõe-se de quatro partes essenciais:

1. **Modelo** – associação das fontes de dados aos respectivos procedimentos de modelação e ligação destes aos protótipos, que efectuam a sua interpretação;
2. **Procedimentos de modelação** – regras de produção baseadas nos sistemas L geoespaciais;
3. **Protótipos** – modelos X3D parametrizados para interpretação das cadeias geoespaciais;
4. **Fontes de dados** – configuração das fontes de dados que servirão como informação inicial dos processos de modelação.

O XML Navi é, na sua essência, um editor visual de XML construído sobre Eclipse RCP e UIForms (Figura 7), mas personalizado para lidar com as especificações de modelação XL3D, através de um conjunto de vistas e uma adaptação ao seu *schema*.

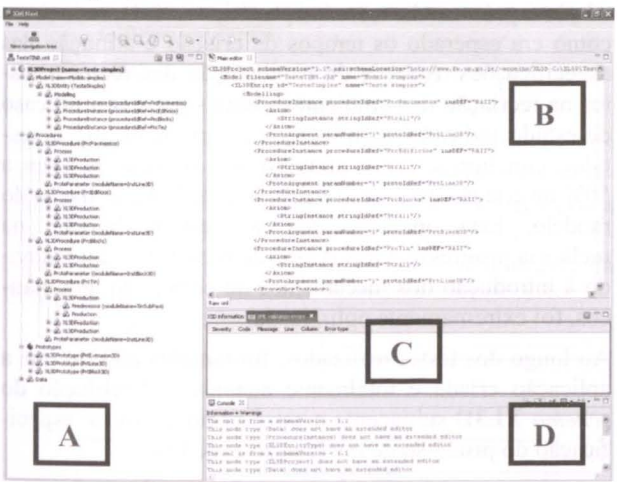

Figura 7 - XML Navi

A ferramenta disponibiliza uma árvore para exploração de documentos XL3D (A) a partir da qual é possível editar, selectivamente, os vários elementos XML (B). A edição de cada elemento na área (B) segue a seguinte filosofia:

- Todos os nós usufruem de um editor XML textual com capacidades avançadas como *syntax highlighting* e outras facilidades típicas como *copy & paste*, *undo/redo*, etc;
- Caso o tipo do elemento disponha de um formulário de edição personalizado e enriquecido para facilitar a manutenção da informação associada, este aparece como um novo *tab* de edição alternativo;
- Se se tratar de um nó representativo de um protótipo XL3D, é visível uma pré-visualização do mesmo como um *tab* de edição alternativo.

Para a edição dos sistemas L geoespaciais, tida como um dos processos mais complexos e morosos, foi desenvolvido um formulário de edição específico que é descrito na secção seguinte.

Na área (C) estão disponíveis dois painéis orientados para a escrita e validação do documento. O primeiro disponibiliza meta-informações do *schema* acerca do elemento da árvore actualmente seleccionado, como por exemplo quais os seus atributos e elementos filho válidos. O segundo painel consiste numa lista de erros de validação do documento a partir da qual é possível localizar as origens do erro. A área (D) é dedicada à escrita de mensagens (informações, avisos, erros) resultantes da reutilização da consola de mensagens do Eclipse IDE.

Foi tido um especial cuidado ao nível da interação com o utilizador, permitindo a edição de vários projectos em simultâneo com uma sincronização total entre os vários painéis de informação (árvore, editores, lista de erros sintáticos, informação detalhada sobre o *schema*). Para além disso, por forma a acelerar a escrita de novos documentos, implementou-se um esquema de templates que, aquando da criação de novos elementos, permite ao utilizador usar informação presente em ficheiros auxiliares, de forma automática.

3.2 Tradução automática das regras de produção dos sistemas L geoespaciais

As regras de produção, embora obedeçam a uma notação própria dos sistemas L geoespaciais, têm que estar representadas em XML de acordo com o *schema* do XL3D, para que o modelador as consiga interpretar.

A notação utilizada para a especificação das regras de produção é a dada por:

$$n: \quad E < P > D : cond \xrightarrow{prob} S_1 ... S_m \quad (1)$$

Onde, P representa o módulo a identificar e a substituir por um ou vários módulos sucessores S; E representa os módulos vizinhos situados à esquerda; D os módulos vizinhos situados à direita, *cond* é a condição da qual depende a aplicabilidade da regra, *prob* a probabilidade da regra de produção e n o número da regra de produção.

A representação via XML de uma gramática constitui um desafio de complexidade considerável. A regra de produção indicada a título de exemplo na figura 8, equivale a 35 linhas de XML, complexo de escrever manualmente e ainda de dificuldade de manutenção crescente à medida que se torna necessário afinar o processo de modelação.

```
3:PaviF1(gPolygon, id, actualIndex, numPoints,
pointsIndex) : actualIndex = numPoints →
   [InstLine3D(gPolygon, pointsIndex, "0 1 0")];
```

Figura 8 – Exemplo de regra de produção

Num processo de modelação real, facilmente se desenvolvem regras (ricas em expressões algébricas) que atingem as centenas de linhas XML, que a cada afinação têm que ser manualmente revistas e corrigidas. Rapidamente se chega à conclusão que, sendo esta a parte mais importante do ponto de vista de autoria, é impraticável a utilização directa do XML.

Relativamente à conversão automática das regras de produção para XML, a primeira conclusão é que seria necessário implementar um *parser*. No entanto, para que, à semelhança dos *schemas*, a ferramenta se adaptasse instantaneamente a alterações na gramática subjacente à notação das regras de produção, este *parser* teria que ser gerado automaticamente.

O Java Compiler Compiler é um gerador de *parsers* (o mais popular) para integração com aplicações Java e, quando utilizado em conjunto com a ferramenta JJTree, permite obter a árvore sintática resultante da operação de *parsing*.

Com algum código adicional, baseado numa versão do *Visitor pattern*, foi definida uma solução para a conversão automática dos valores de texto relevantes, ao nível da árvore síntatica, para XML (Figura 9).

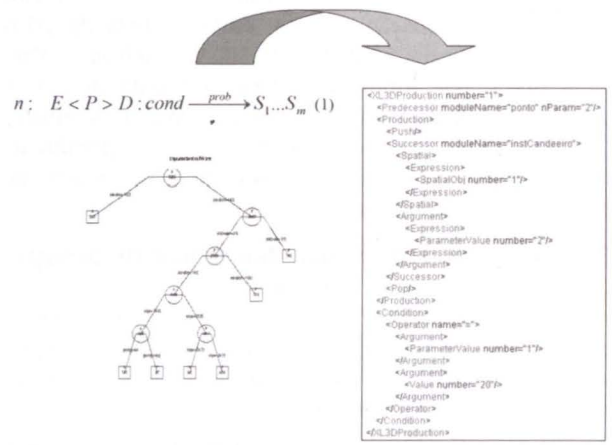

Figura 9 - Conversão automática de regras de produção

3.3 Adaptabilidade do processo de desenvolvimento

A determinação inicial foi desenvolver esta ferramenta de forma a que fosse capaz de se adaptar a qualquer XML-Schema durante a fase de desenvolvimento do modelador XL3D, sem que o código da aplicação tivesse que sofrer alterações.

Este requisito garante total isolamento do XML-Schema relativamente à evolução da especificação XL3D, assim como garante a utilização da ferramenta para outros XML-Schemas que venham a ser criados no âmbito deste projecto (ou outros). A tecnologia que o tornou possível foi a *framework* de "*data binding*" XMLBeans que permite gerar automaticamente bibliotecas de interfaces e classes Java a partir de *schemas*.

Recorrendo a três APIs presentes nestas bibliotecas, qualquer ficheiro XML pode ser manipulado, de forma dinâmica, sem que a sua estrutura tenha que ser conhecida *à priori*:

- **XmlObject** – classes que mapeiam directamente os dados contidos no ficheiro XML;
- **XmlCursor** – classes que permitem o acesso ao XML praticamente ao nível do *parser*;
- **SchemaType** – classes que reflectem o próprio *schema* e que permitem o acesso à meta-informação.

A utilização destas três APIs permite determinar, em tempo de execução, toda a informação necessária, reduzindo o impacto das alterações nos *schemas* a uma mera (re)geração da(s) biblioteca(s), processo automático que decorre em poucos minutos.

3.4 Resultados

Por forma a testar a estabilidade da aplicação XML Navi e a eficácia do mecanismo de tradução das regras de produção, foram realizados diversos testes, de acordo com o caso de estudo definido em [Coelho06b]. Foram recriados os diversos ambientes virtuais produzidos, e realizada a conversão da totalidade dos sistemas L geoespaciais definidos.

Relativamente à conversão automática das regras de produção, a taxa de sucesso alcançada foi de 100%, após mais de uma centena de regras terem sido convertidas e os resultados obtidos serem idênticos ao XML manualmente escrito no referido caso de estudo.

Esta mais valia, acrescida das capacidades extendidas de edição e pré-visualização dos protótipos X3D, reduziram como era esperado os tempos de criação e afinação das especificações. Através da comparação dos tempos gastos na recriação dos ambientes virtuais presentes no caso de estudo indicado, com os tempos originalmente investidos, concluiu-se ter-se atingido uma redução superior a 50% na criação da especificação e de 80% na afinação do modelo. Esta operação consiste, essencialmente, na melhoria progressiva das regras de produção e que, devido à introdução dos mecanismos de conversão automáticos, foi extremamente optimizada.

Ao longo dos testes realizados, foi também aferido que a aplicação criada é totalmente adaptável à evolução do *schema* XL3D subjacente a estes documentos de especificação do processo de modelação expedita.

4. APLICAÇÃO DE VISUALIZAÇÃO INTERACTIVA DE AMBIENTES URBANOS VIRTUAIS

Complementarmente à ferramenta de autoria foi desenvolvida uma aplicação para a visualização interactiva dos modelos X3D gerados a partir das especificações criadas. Esta baseou-se nos mesmos conceitos de prototipagem rápida recorrendo à RCP Eclipse.

A interface desta aplicação protótipo (Figura 10) é funcionalmente composta por 3 componentes principais:

- Um *browser* que faz a visualização da cena 3D **(A)**;

- Um painel que permite a visualização de mapas 2D auxiliares à navegação na cena **(B)**;
- Um painel com controlos para testar as capacidades de interacção em tempo real **(C)**.

Figura 10 - X3D Navi

A decisão de se avançar para a construção de uma ferramenta de visualização deve-se às limitações encontradas ao nível da interação exterior com a cena X3D. No estado actual desta tecnologia, os modelos são normalmente visualizados recorrendo a um *plugin* para os programas de navegação Web comerciais. Neste ambiente uma cena pode tornar-se interactiva de duas formas:

- *scripts* ECMA internos que dotam a cena de comportamentos;
- classes Java que interagem com a cena e são invocadas a partir desta.

No entanto, em paradigmas de interacção complexos, nos quais a iniciativa do acesso possa ser exterior, estas soluções são insuficientes. Nestes casos, torna-se necessária a utilização de um *browser* Java que, integrado numa aplicação, permita a interação directa com a cena através da SAI (Figura 11).

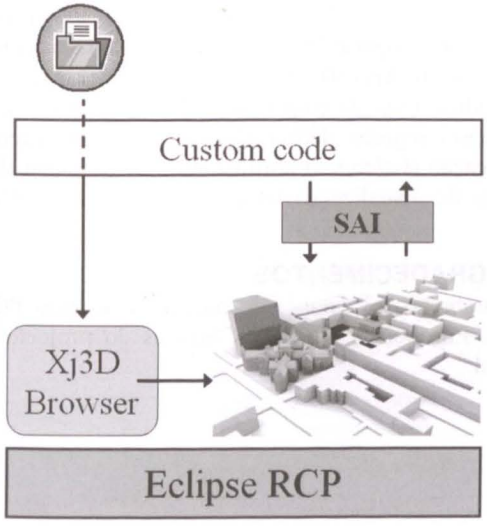

Figura 11 - Arquitectura macro do X3D Navi

A sincronização entre a cena 3D e o mapa 2D foi a funcionalidade mais complexa de atingir, dado que não há nenhuma forma directa de ler as coordenadas e orientação actuais do *avatar* a partir da cena renderizada. Por este motivo, foi necessário criar um método indirecto que consiste na utilização de um sensor do tipo *Proximity* que gera eventos com as informações pretendidas sempre que ocorram alterações na cena.

Após a criação da cena pelo *browser* Java, segue-se uma etapa inicial de configuração e construção de objectos auxiliares para o controlo da interacção, nos quais se inclui este sensor (Figura 12). Este é posicionado de acordo com as informações obtidas de um nó especialmente criado na cena pelo modelador com as meta-informações necessárias (coordenadas centrais da cena, ficheiro com o mapa 2D correspondente, etc).

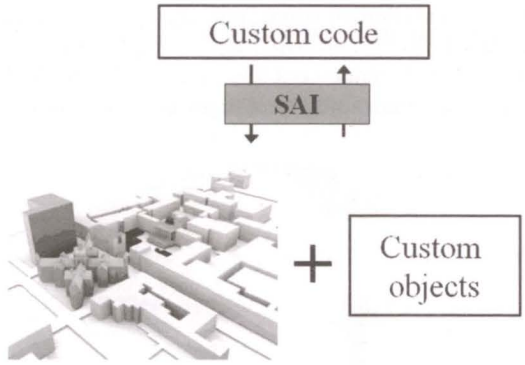

Figura 12 - Detalhe da arquitectura do X3D Navi

Quando estas informações são correctamente introduzidas pelo modelador no ficheiro X3D, a ferramenta de visualização mostra o mapa 2D no qual é possível acompanhar os movimentos do *avatar* no plano XZ (círculo vermelho) e a correspondente orientação (ponto verde) (Figura 13).

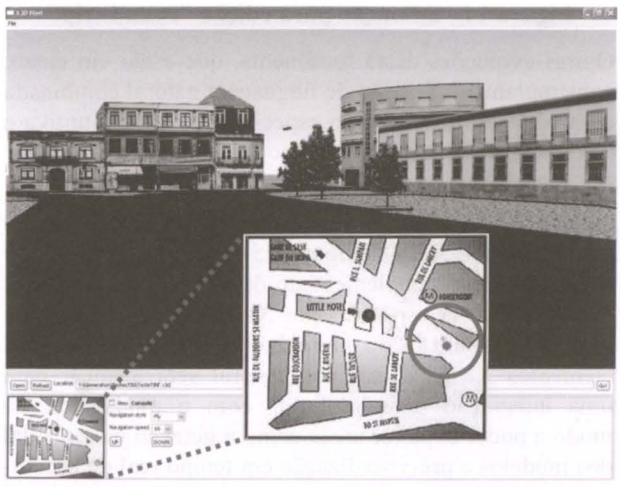

Figura 13 - Detalhe do mapa auxiliar 2D

5. CONCLUSÕES E TRABALHO FUTURO

A modelação expedita permite acelerar o processo de geração de ambientes urbanos virtuais mas requer a utilização de documentos de especificação, cuja criação se reveste de dificuldades. Neste tipo de sistemas, cada vez

mais complexos e tecnologicamente heterógeneos, a criação/utilização de ferramentas que permitam simplificar estes processos e focar os autores no cerne dos problemas é determinante.

Neste artigo, apresentou-se o XML Navi, uma ferramenta que através da conjugação de várias tecnologias, permite a criação e manutenção, de forma fácil e intuitiva, dos documentos que suportam a geração de ambientes virtuais urbanos segundo a tecnologia XL3D.

Escrever as regras de produção numa notação natural, legível e conforme com a literatura da área, simplifica não só a elaboração dos processos de modelação, mas também a passagem de conhecimento. As horas ou mesmo dias perdidos na conversão manual das regras de produção para uma estrutura XML, reconhecida pelo modelador, foram suprimidas. Neste capítulo, existe ainda espaço para melhorias como, por exemplo, a implementação da edição "*round trip*", isto é, implementar a conversão inversa fazendo com que as alterações no XML se reflictam automaticamente na regra de produção (Figura 14).

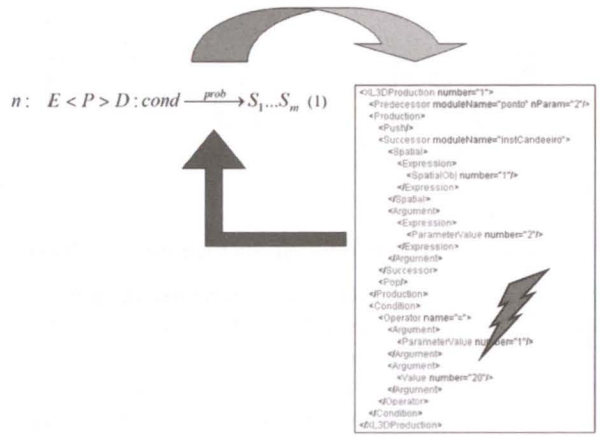

Figura 14 - Evolução para edição "round trip"

Outras evoluções desta ferramenta, que estão em curso, contemplam a utilização de linguagem natural combinada com editores 3D para uma especificação mais intuitiva e automatizada dos sistemas L geoespaciais.

Um dos principais trabalhos a desenvolver no futuro será a evolução do XML Navi para um verdadeiro *front-end* do modelador XL3D (Figura 15). Embora o protótipo actual tenha sido implementado de forma a ser adaptável às alterações na especificação XL3D, na gramática das regras de produção ou nos editores personalizados dos elementos XML, o caminho natural será avançar para uma integração mais profunda com o modelador, de modo a poder explorar áreas como a geração incremental dos modelos e pré-visualização em tempo real de processos de modelação.

A modelação incremental reveste-se de particular relevância na fase de desenvolvimento de especificações, permitindo visualizar a modelação de partes do modelo global, durante fases intermédias deste processo.

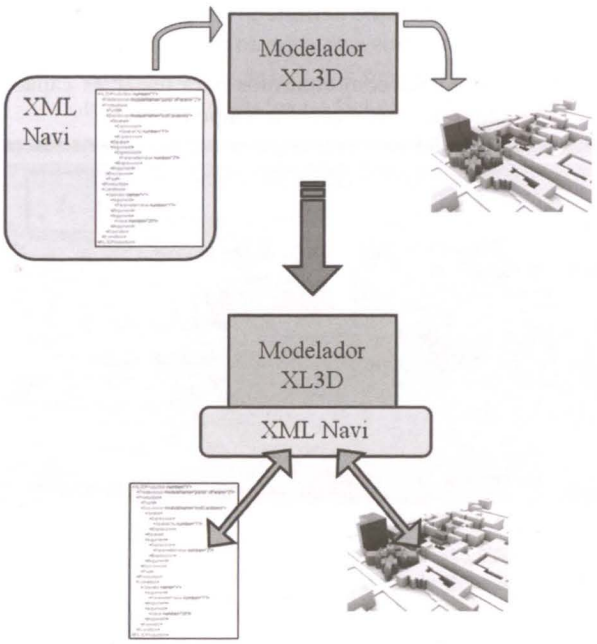

Figura 15 - Evolução do XML Navi para front-end

Seguindo a mesma filosofia de prototipagem rápida, foi também desenvolvida uma aplicação para visualização interactiva de ambientes urbanos virtuais (X3D Navi), que possibilita o desenvolvimento futuro de novas metáforas de interacção sobre estes ambientes. Esta aplicação reutiliza o *browser* Java Xj3D e tira partido da API SAI para permitir uma interação em tempo real com a cena, provando que, em fases mais avançadas, será possível manipular o modelo a partir do exterior.

Concluiu-se também que as tecnologias de visualização, nomeadamente o *browser* Xj3D, ainda apresentam muitas limitações. A API da Scene Access Interface é pouco orientada a objectos, tornando o código complexo e moroso de escrever. O comportamento desta API apresentou também alguma instabilidade, o que obrigou a reforços constantes no código de modo a lidar com problemas de precisão de vírgula flutuante, lançamento de excepções e reacções "estranhas" caso as ordens de determinadas operações fossem alteradas. O recente aparecimento do Ajax3D [Ajax3D] associado principalmente ao produto Flux da empresa Media Machines é também um facto a registar, dado poder viabilizar uma alternativa à aplicação *desktop*, simplificando a distribuição de programas de visualização baseados unicamente em *browsers*.

6. AGRADECIMENTOS

Este trabalho foi financiado parcialmente pelo POSI, a União Europeia e o FEDER através do projecto Porto Digital.

7. REFERÊNCIAS

[Ajax3D] The Open Platform for Rich 3D Web Applications. < http://www.ajax3d.org/ >, Setembro 2007

[Boehm03] Boehm, B.; Turner, R.; Observations on Balancing Discipline and Agility, Proceedings of Agile Development Conference, pp. 32-39, 2003.

[Coelho03] Coelho A., Sousa A., Ferreira F.: 3D Modelling of Large Urban Scenes from Diverse Sources of Information. Proceedings of the 7th ICCC/IFIP International Conference on Electronic Publishing, pp. 278-287, 2003.

[Coelho04] Coelho, A.; Sousa, A.; Ferreira, F.; Expeditious Modelling of Urban Scenes, Revista VIRtual (ISSN: 0873-1837, http://virtual.inesc.pt/) - edição especial "Advances in Computer Graphics in Portugal", 2004.

[Coelho06a] Coelho, A.; Bessa, M.; Sousa, A.; Ferreira, F.; Expeditious Modelling of Virtual Urban Environment with Geospatial L-systems, Proceedings of SIACG 2006 – Ibero –American Symposium in Computer Graphics, pp. 109-119, 2006.

[Coelho06b] Coelho, A.; Modelação Expedita de Ambientes Urbanos Virtuais Baseada em Interoperabilidade e Contextualização Geoespacial, Tese de Doutoramento, FEUP, 2006.

[Figueroa05] – Figueroa, P.; Medina, O.; Jiménez, R.; Martínez J.; Albarracín, C.; Extensions for Interactivity and Retargeting in X3D, Proceedings of Conf. Web 3D Technology, pp. 103-110, 2005.

[Leavitt06] Leavitt, N.; Browsing de 3D Web, IEEE Computer Magazine, pp. 18-21, 2006

[LeBlanc05] Le Blanc, A.; Bunt, J.; Kwok, Y.; Petch, J.;The Virtual Learning Space - An Interactive 3D Environment, Proceedings of Conf. Web 3D Technology, pp. 93-102, 2005.

[RCPEclipse] Rich Client Platform. < http://wiki.eclipse.org/index.php/Rich_Client_Platform>, Setembro 2007

[Vitzthum05] Vitzthum, A.; Pleuß, A.; SSIML: Designing Structure and Application Integration of 3D Scenes, Proceedings of Conf. Web 3D Technology, pp. 9-17, 2005.

[Web3D] The Web3D Consortium. <http://www.web3d.org/>, Setembro 2007

[X3D] Web3D Consortium: "X3D: The Virtual Reality Modelling Language – International Standard ISO/IEC 14772:200x". < http://www.web3d.org/x3d/specifications/#x3d-spec>, Setembro 2007

[XAmple] Java XML Editor - GUI Generator. <http://www.felixgolubov.com/XMLEditor/>, Setembro 2007

[Xeena] alphaWorks : Xeena : Overview. <http://www.alphaworks.ibm.com/tech/xeena>, Setembro 2007

[Xj3D] The Xj3D Project. <http://www.xj3d.org/>, Setembro 2007

[XML-Schema] XML-Schema. <http://www.w3.org/XML/Schema>, Setembro 2007

[XMLSpy] XML Editor from Altova: XMLSpy. <http://www.altova.com/XMLSpy>, Setembro 2007