VISUALIZACIÓN DE GRAFOS EN EL CONTEXTO DEL MODELO UNIFICADO DE VISUALIZACIÓN[1]

Sebastián Escarza Silvia Castro Sergio Martig

Departamento de Ciencias e Ingeniería de la Computación
Laboratorio de Investigación en Visualización y Computación Gráfica
Instituto de Investigación en Ciencia y Tecnología Informática (IICyTI)
Universidad Nacional del Sur
{se, smc, srm}@cs.uns.edu.ar

Resumen

Los grafos constituyen un formalismo muy versátil en el modelado y la representación de diversos dominios de aplicación. La visualización de los mismos es un área de gran actividad dentro de la Visualización de Información. Dada la heterogeneidad de dominios en los que se los utiliza, la identificación de características comunes a dichos dominios sumada a problemas propios del manejo de grandes volúmenes de información hace que su abordaje constituya un desafío. En este escenario, el planteo de modelos de referencia que unifiquen estos aspectos, como el Modelo Unificado de Visualización, permiten factorizar características comunes y ubicarlas en un marco que posibilite su estudio y aplicación de una manera más efectiva en las visualizaciones. El presente trabajo tiene como objetivo principal evaluar la aplicabilidad de dicho modelo a una situación concreta así como identificar operadores y operandos propios del área de Visualización de Grafos de manera de contribuir con la validación del Modelo Unificado de Visualización. Para ello se implementó un prototipo de herramienta de visualización de grafos.

Palabras clave: Visualización de Grafos – Visualización de Información - Visualización

Introducción

La Visualización de Grafos es un área de intensa actividad en materia de investigación en estos últimos años. La gran cantidad de aplicaciones existentes colocan a los grafos como un formalismo muy versátil en el modelado y la representación de diversos dominios de aplicación. Esta diversidad de áreas dificulta la identificación de características comunes a todas y como consecuencia de ello el problema de construir visualizaciones útiles en todos los dominios de aplicación se torna complejo.

Sumado a lo anterior existe el problema de la cardinalidad de los conjuntos de datos existentes que puede exceder incluso la cantidad de píxeles de la pantalla. Representar grafos con una gran cantidad de nodos y arcos aún no dispone de una solución general, válida para todos los casos y dominios de aplicación, que se destaque por sobre las demás. La disposición de elementos en pantalla debe seguir criterios estéticos y prácticos no siempre claramente definidos, para resultar en una representación útil al usuario. Además ciertos enfoques que redundan en visualizaciones útiles en un determinado dominio de aplicación suelen no ser los más adecuados en otras áreas. El problema se agrava notablemente al representar visualmente no sólo la topología de un grafo sino también la información asociada a sus nodos y arcos.

En este contexto resulta crucial que el usuario interactúe con la representación de manera ágil para adaptarla a sus necesidades particulares. Lejos quedó el tiempo en que la vista de la información se limitaba a una imagen estática en pantalla. Actualmente las posibilidades de exploración de los datos y la personalización de la vista constituyen características invalorables desde el punto de vista de la utilidad de la visualización. Sin embargo este aspecto agrega dos componentes más a la complejidad del problema de visualizar grafos: la identificación de interacciones adecuadas a los

[1] El presente trabajo fue parcialmente financiado por la Universidad Nacional del Sur, Bahía Blanca, Argentina y el PICT 2003 Nº 15043 (Agencia Nacional de Promoción Científica y Tecnológica).

fines del usuario y el logro de tiempos de respuesta interactivos por parte de la visualización para posibilitar una exploración ágil de los mismos.

Objetivos

Dada la heterogeneidad existente entre los dominios de aplicación de la Visualización de Grafos y la diversidad de técnicas, es necesario factorizar características comunes y ubicarlas en un marco que permita su estudio y aplicación de una manera efectiva en las visualizaciones.

El Modelo Unificado de Visualización (MUV) [MCFE03] proporciona un marco conceptual que permite ubicar los procesos y los estados intermedios de los datos y definir las interacciones explícitamente. Dicho modelo intenta unificar los aspectos propios a la problemática de visualizar datos y trasciende la Visualización de Grafos. Presenta una arquitectura de pipeline de visualización en la cual todas las etapas (estados intermedios de los datos) y las transformaciones (de los datos) entre dichas etapas están definidas. Los diferentes dominios de aplicación presentan operandos y operadores de "alto nivel de abstracción" que les son propios. El desafío consiste en validar la completitud de los operadores de "bajo nivel de abstracción" propuestos en el Modelo Unificado de Visualización para soportar los requerimientos de cada dominio particular.

El presente trabajo tiene como objetivo principal evaluar la aplicabilidad de dicho modelo a una situación concreta así como identificar operadores y operandos propios del área de Visualización de Grafos de manera de contribuir a la validación del modelo.

Adicionalmente, este trabajo permitirá obtener una visión cercana y concreta de los problemas principales del área y analizar posibles soluciones o aproximaciones de solución que posibiliten la generación de visualizaciones efectivas.

Se decidió la implementación de una herramienta de visualización de grafos para poder evaluar los aspectos citados anteriormente. Dicha herramienta no pretende convertirse en una aplicación de producción sino que fue concebida con propósitos de evaluación exclusivamente. La construcción de la herramienta se encuentra actualmente en un estado avanzado y permite obtener una visión de ciertos aspectos del problema que sólo salen a la luz durante el desarrollo y uso de la misma. Dicha construcción ha sido incremental permitiendo obtener una versión ejecutable de funcionalidad mínima en etapas iniciales del desarrollo; gracias a esto, muchas conclusiones obtenidas inicialmente se han traducido en mejoras a la aplicación. Dichas mejoras permiten obtener nuevas conclusiones sobre la base de una aplicación de mayor calidad y funcionalidad.

A lo largo del trabajo se hará un paralelo entre las etapas del proceso de visualización implementado en la aplicación y los estados intermedios y transformaciones del Modelo Unificado de Visualización. Además se tratarán, de manera particular, los problemas más relevantes surgidos durante el desarrollo y el uso de la aplicación.

El Modelo Unificado de Visualización

El desarrollo de un modelo de referencia que unifique los conceptos de las distintas áreas de la Visualización se ve motivado por la necesidad de establecer un marco común para que ***los usuarios*** provenientes de distintos dominios de aplicación logren interacciones efectivas basándose en un único modelo mental y ***los diseñadores*** de visualizaciones expresen claramente las

transformaciones que deben sufrir los datos hasta la obtención de la vista, las operaciones que deben proveerse y la manera en que debe interactuarse sobre la visualización. El Modelo Unificado de Visualización [MCFE03], en este sentido, establece un marco conceptual en términos del cual definir estados intermedios, transformaciones, operadores, operandos e interacciones de manera independiente de todo dominio de aplicación.

En este contexto el Modelo Unificado de Visualización [MCFE03] constituye un modelo que es consistente con las posibles intenciones de los usuarios y brinda un marco en el que es claro cómo interactúan las posibles transformaciones y operaciones provistas. Por otro lado también beneficia a los diseñadores al momento de extender un sistema de visualización para incorporar nuevos dominios de aplicación, brindando un marco de referencia que les define cuáles son las transformaciones que deben sufrir los datos y el conjunto básico de operaciones que se deberían proveer.

Además, la separación en etapas del proceso, posibilita asociar ciertos operadores a determinadas instancias de dicho proceso, permitiendo que procesos de visualización de dominios de aplicación diferentes que comparten estados intermedios queden modelados de manera uniforme.

Como se dijo antes, el Modelo Unificado de Visualización [MCFE03] define un pipeline de visualización en el cual se modelan las transformaciones que sufren los datos del dominio de aplicación desde su origen hasta la obtención de la vista. Dicha vista constituye el punto de partida para que el usuario interactúe con la visualización y la adapte a sus necesidades.

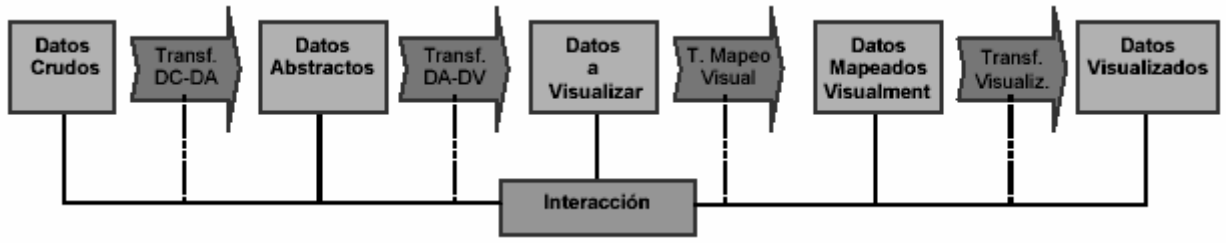

Fig. 1: El Modelo Unificado de Visualización

Las etapas del modelo se detallan a continuación:

- **Datos crudos**: Los datos provenientes del dominio de aplicación.

⇒ **Transformación de Datos Crudos en Datos Abstractos**: Permite al usuario seleccionar los datos que llevará a un formato manejable por la aplicación para su posterior visualización. Se pueden generar metadatos.

- **Datos Abstractos**: Son los datos potencialmente visualizables, compuesto por datos y metadatos.

⇒ **Transformación de Datos Abstractos en Datos a Visualizar**: Permite seleccionar el subconjunto de los datos abstractos que se visualizará.

- **Datos a Visualizar**: Son todos los datos que van a estar presentes en la visualización.

⇒ **Transformación de Mapeo Visual**: Permite especificar cómo serán visualizados los Datos a Visualizar. Se define el sustrato espacial de la visualización, los elementos visuales a utilizar y sus atributos y para cada uno de ellos a que atributos de los datos se asociarán.

- ☐ **Datos Mapeados Visualmente**: Son los datos a visualizar enriquecidos con información sobre cómo representarlos en pantalla.

- ⇒ **Transformación de Visualización**: Constituye la aplicación de una técnica de visualización que soporte el mapeo visual definido anteriormente.

- ☐ **Datos Visualizados**: Son los datos presentados en pantalla. Es el punto inicial para la exploración del usuario del conjunto de datos.

- ❖ **Interacción**: El usuario desde los Datos Visualizados afecta la visualización para adaptarla a sus necesidades. La forma en que interactúe determinará qué etapas del pipeline se involucran para resolver dicha interacción.

El modelo propone un conjunto de operadores y operandos denominados de bajo nivel; aplicables a las distintas etapas y transformaciones, así como también un conjunto de interacciones generales de alto nivel características de todo proceso de visualización.

La diversidad de los dominios de aplicación y la consiguiente proliferación de técnicas de visualización, determina que cada área defina sus propios operandos y operadores. Este hecho hace que resulte necesario evaluar la completitud de los operadores y operandos propuestos en el Modelo Unificado de Visualización. Contar con una aplicación concreta en un área tan compleja como la Visualización de Grafos es de gran valor para lograr este objetivo.

La Visualización de Grafos en el contexto del MUV

A continuación se presenta un caso concreto de visualización de grafos a partir del cual es posible validar el Modelo Unificado de Visualización en los aspectos comentados anteriormente. En las secciones siguientes se describe una implementación particular y luego se procede a establecer el paralelo existente entre la arquitectura de dicha implementación y la planteada en el MUV.

El pipeline de visualización de grafos de GVL/VTK

En esta sección se describe un caso concreto de una implementación de un pipeline de visualización de grafos. Las etapas presentes en el mismo son representativas de una aplicación básica del área y se originan en las facilidades provistas por la Graph Visualization Library (GVL) [VTKGvl], una librería que incorpora soporte para grafos al Visualization Toolkit [VTK]. La correlación al implementar este pipe en el contexto del Modelo Unificado de Visualización permitirá ver cómo el diseñador de aplicación verá facilitada su tarea en un caso concreto para un dominio de aplicación particular facilitándole la identificación de operadores e interacciones propias del área.

El pipeline de visualización implementado se configura dinámicamente y constituye el núcleo del prototipo. Las etapas iniciales de dicho pipeline se centran en las facilidades provistas por la Graph Visualization Library [VTKGvl] para la lectura y manipulación de grafos. Una vez que se obtiene la representación geométrica (ya sea en 2D o 3D) de dicho grafo, el pipeline utiliza las clases de VTK [VTK] para el procesamiento final.

La Graph Visualization Library [VTKGvl] adiciona a VTK un nuevo objeto de datos: el vtkGraph. Dicho objeto mantiene los nodos y arcos de un grafo en particular asignando identificadores a los mismos de manera de poder referenciarlos. Además, provee una interfaz de alto nivel para manipular el grafo que permite su recorrido mediante iteradotes y el agregado y eliminación de

nodos y arcos, entre otras características. A dicho objeto de datos se le incorpora información adicional (métricas o información posicional de nodos, por ej.) conforme circula por las diferentes etapas del pipeline de visualización.

El pipeline de visualización se origina con una etapa de ***lectura de archivos***. Allí archivos de tipo gml son leídos y su contenido (el grafo) es representado por un vtkGraph.

Una vez hecho esto existe una etapa opcional de ***cómputo de métricas*** sobre los nodos del grafo. Dichas métricas podrían utilizarse entre otras posibles tareas para realizar filtrados sobre el grafo. La Graph Visualization Library provee la métrica Strahler (para árboles) y una generalización de la misma a grafos (utilizada por el prototipo). Para modelar la noción de localidad o vecindad de un nodo, se implementó una métrica de localidad (basada en la distancia que existe entre el nodo seleccionado y sus vecinos) y se incluyó en el pipeline del prototipo. Dicha métrica posibilita el filtrado de los nodos más lejanos y permite al usuario navegar por el grafo visualizando sólo cierto nodo y su contexto. Una vez concluida esta etapa se adiciona al vtkGraph información que asocia un valor para la o las métricas elegidas con cada nodo del grafo que representa.

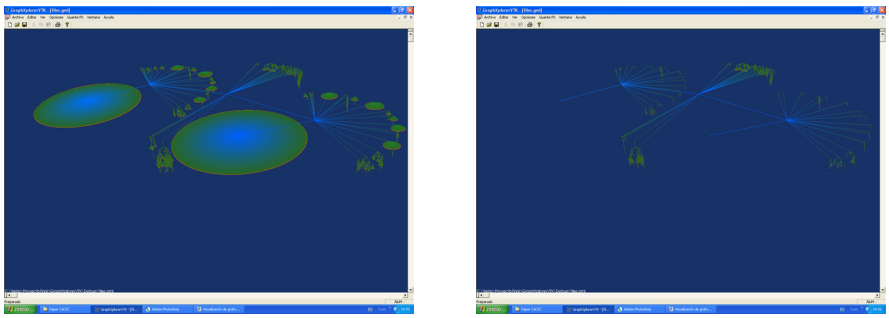

Fig. 2: Los filtrados se basan en métricas y son un recurso importante ante la complejidad que presentan los grafos de grandes cardinalidades. Las imágenes corresponden a un grafo sin filtrar (izquierda) y luego de aplicarle un filtro basado en la métrica de Strahler (derecha). También se puede apreciar cómo esta métrica captura la estructura del grafo representado.

Posteriormente al cálculo de las métricas sigue la etapa (también opcional) de ***filtrados***. En esta, en base a las métricas seleccionadas, se determina qué subgrafo se visualizará. Se selecciona un subconjunto de los nodos del grafo original mediante la métrica calculada y se mantienen aquellos arcos cuyos nodos asociados (ambos) hayan sido seleccionados. La Graph Visualization Library [GVLVtk] provee filtros que permiten elegir los n mejores nodos del grafo, o los p% mejores nodos del grafo respecto a la métrica calculada o bien todos los nodos del grafo cuyo valor para la métrica calculada se encuentre entre determinados valores.

Una vez hecho esto se le aplica un ***algoritmo de layout*** al grafo en cuestión. En esta etapa se asocia información posicional a cada nodo del grafo (ie. se agrega al vtkGraph). La Graph Visualization Library [GVLVtk] provee layouts basados en árboles de cubrimiento de grafos. Se extrae un árbol de cubrimiento al grafo (posiblemente eliminando algunos arcos), se le aplica un algoritmo de layout de árboles y se añaden a la representación los arcos eliminados. Además, implementa el algoritmo Gem 2D de Arne Frick, el cual es dirigido por fuerzas y computacionalmente costoso y el algortimo HDE (Higher-Dimensional Embbeding) de Harel y Koren's. En este último se dibuja el grafo en un espacio de alta dimensionalidad y luego se proyecta a un espacio 2D o 3D.

Así, se logra un grafo con información espacial asociada. La Graph Visualization Library [GVLVtk] provee dos filtros para ***extraer la geometría*** de los nodos y los arcos del grafo. Los nodos son extraídos como un conjunto de puntos en el espacio y los arcos como un conjunto de líneas (rectas o posiblemente poligonales). A partir de este instante se obtienen entidades vtkPolyData manejables por VTK [VTK]. Lo interesante de esto es que aquí podría utilizarse cualquier facilidad provista por VTK [VTK] para enriquecer la visualización.

Posteriormente el prototipo permite ***determinar los elementos visuales*** asociados a nodos y arcos del grafo. Se permite seleccionar los elementos visuales con los que se representarán los nodos (puntos, cubos, esferas, etc.) y los arcos (líneas, tubos o cintas).

Luego de definir la geometría completa de la representación se procede a realizar el ***mapeo de los grafos a primitivas gráficas***. En esta etapa se definen los mapeadores de VTK [VTK] para esta tarea y se ***asocian*** tablas de ***color y transparencia*** para determinar en base a qué atributo de los grafos será asociado el color y la transparencia (función transferencia). La herramienta permite seleccionar el mapeo que se utilizará para reflejar color y transparencia en función de la métrica seleccionada para el grafo que se intenta visualizar. Resulta de gran ayuda al usuario dejar la porción del grafo que se constituye como su foco de atención opaco y gradualmente hacer tender a transparente su contexto hasta hacer invisibles aquellas partes del grafo más alejadas.

Luego se definen los actores que participarán de la escena y se determina el mecanismo y la ventana de destino para el renderizado así como el estilo de interacción con la vista (pasos típicos finales de cualquier pipeline de visualización de VTK [VTK]).

Relación entre el pipeline de GVL/VTK y el del MUV

En esta sección se verá cómo fue establecida la relación que existe entre el pipeline de visualización implementado y el pipeline propuesto por el Modelo Unificado de Visualización [MCFE03]. De este modo se aprecia cómo el Modelo Unificado de Visualización se constituye en un marco de referencia natural para este caso concreto de Visualización de Grafos.

Los archivos gml que la Graph Visualization Library [GVLVtk] es capaz de leer se asimilan a los ***Datos Crudos*** del pipeline del Modelo Unificado de Visualización, ya que los Datos Crudos constituyen aquellos datos provenientes del dominio de aplicación y se encuentran en un formato no manejable directamente por la aplicación. Los archivos gml no son directamente manejables por la herramienta. Dichos archivos se convierten a objetos vtkGraph para su procesamiento.

La ***Transformación de Datos Crudos en Datos Abstractos*** consiste en la simple lectura del archivo gml y la generación del objeto vtkGraph por parte del "reader" provisto por la Graph Visualization Library [GVLVtk].

El objeto vtkGraph generado a partir de la lectura desde el archivo, así como el objeto vtkGraph al cual se le adicionó información acerca de una o varias métricas sobre sus nodos constituyen el estado de los ***Datos Abstractos***. Dichos datos abstractos son datos potencialmente visualizables por el usuario y en un formato manejable por la aplicación.

La ***Transformación de Datos Abstractos en Datos a Visualizar*** se produce en la etapa de filtrado del grafo (en base a la métrica elegida). El usuario de la herramienta determina en esta etapa qué subgrafo del grafo será visualizado.

El subgrafo obtenido luego de la etapa de filtrado constituye el conjunto de ***Datos a Visualizar***. Todos los nodos y arcos presentes en este subgrafo serán representados en la vista de alguna manera.

La ***Transformación de Mapeo Visual*** permite especificar al usuario cómo quiere visualizar los datos presentes en el conjunto de Datos a Visualizar. Esta etapa del pipeline del Modelo Unificado de Visualización se asocia a varias etapas del pipeline de la herramienta. Inicialmente se especifica el sustrato espacial del grafo al aplicar un algoritmo de layout; posteriormente y teniendo ya la

geometría de los vértices y arcos se especifican los elementos visuales (qué glifo 3D se aplicará a los nodos y qué primitiva a los arcos). Finalmente durante la actividad de los mapeadores de VTK [VTK] se asocian atributos gráficos como el color y la transparencia a los valores (métricas) asociados a los nodos y arcos.

Luego del procesamiento realizado por los mapeadores de VTK [VTK], los datos (el grafo) quedan mapeados visualmente (Datos Mapeados Visualmente en el pipeline del MUV). Este estado de los datos no se refleja en ningún dataset y el programador de VTK [VTK] no tiene acceso a la representación ya que VTK [VTK] los maneja internamente (en la comunicación entre mapeadores y actores).

La ***Transformación de Visualización*** aplica una técnica de visualización que soporte el mapeo visual presentado. En el caso del prototipo se utiliza una técnica de rendering de polígonos que utiliza OpenGL vía VTK [VTK].

Finalmente, los ***Datos Visualizados*** se presentan de manera bidimensional (pixeles) en pantalla en alguna de las ventanas que la herramienta provee. A partir de esta vista se originan las diversas interacciones (zoom, rotaciones, traslaciones, selección del nodo actual, etc) que permiten al usuario explorar la representación.

Prototipo de Visualización de Grafos

La construcción del prototipo en base al dominio de Grafos se origina en la necesidad de contar con una implementación concreta que permita evaluar la aplicabilidad del Modelo Unificado de Visualización [MCFE03] tal y como se explicara con anterioridad. Además el emprender el desarrollo de una aplicación concreta permite adquirir experiencia en el diseño y construcción de visualizaciones que si bien son específicas del área de Visualización de Grafos, poseen características similares con otros dominios. Por otro lado se dispone de una aplicación funcional que permite identificar aspectos y consideraciones desde el punto de vista del usuario de una visualización, que sólo surgen con el uso de una aplicación de este tipo.

Características

La implementación del prototipo se hizo utilizando el Microsoft Visual C++ 6.0. Dicho prototipo es una aplicación MFC (utiliza las Microsoft's Fundation Classes) y se implementó utilizando VTK 4.2 [VTK] y la Graph Visualization Library versión 1.10 [GVLVtk].

La aplicación permite visualizar grafos en el formato de archivo gml provisto por la GVL [GVLVtk] e interactuar con las vistas generadas. Como tareas principales es posible realizar filtrados, seleccionar el layout, asociar diferentes elementos geométricos a nodos y arcos, seleccionar nodos para visualizar su contexto (navegación del grafo) y seleccionar el mapeo de color y transparencia para arcos y nodos. Se permiten manejar múltiples vistas de un mismo grafo simultáneamente y tener abiertos múltiples grafos en la misma aplicación.

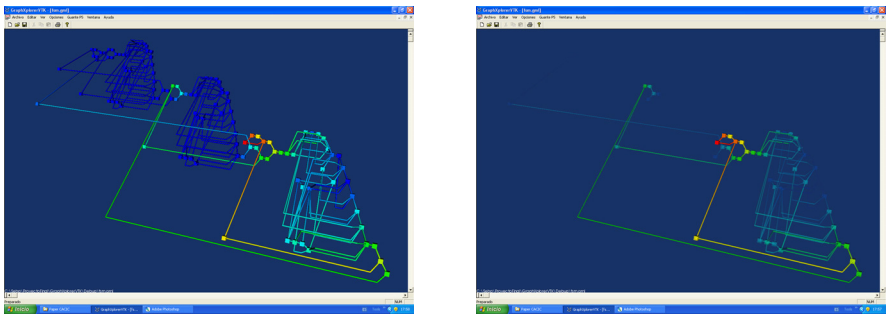

Fig. 3: El uso de la transparencia facilita al usuario la identificación de su foco de atención suavizando el impacto del contexto de exploración en la vista.

En el Modelo Unificado de Visualización [MCFE03] se enfatiza el hecho de que en cada etapa se pueden aplicar varias transformaciones originando diversas instancias de las etapas posteriores. De esta manera se pueden obtener varios Datos Visualizados a partir de los Datos Mapeados Visualmente, o varios Datos Mapeados Visualmente a partir de los Datos a Visualizar por citar dos ejemplos.

En este aspecto la arquitectura documento-vista que provee el framework de las Microsoft Fundation Classes (MFC) resulta en un diseño de la aplicación que permite abordar estas características de manera muy natural. Bajo la arquitectura documento-vista, se modelan los datos de una determinada aplicación (el grafo en este caso) como un documento al cuál se le asocian una o más vistas. Esta separación entre los datos y su representación visual redunda en un diseño más modular y extensible.

En relación a esto, se modeló como parte del documento de la aplicación el vtkGraph obtenido a partir del archivo gml y el nodo actual seleccionado. Las restantes etapas del pipeline se modelaron en la vista. De esta manera el prototipo presentado permite tener varias vistas con, posiblemente, diferentes métricas, filtrados, layouts, elementos y atributos visuales a partir de un mismo grafo. Además el hecho de que el nodo actual seleccionado sea parte del documento implica que será común a todas las vistas logrando de esta manera que al seleccionar un nodo en una de las vistas, esto se refleje en las demás (vinculación de vistas).

Utilizar una librería estándar como VTK [VTK] para la implementación del prototipo (extendida mediante la GVL en lo que hace a modelar y visualizar grafos) facilita el desarrollo ya que mucha funcionalidad propia de las herramientas de visualización en general (y del dominio de grafos mediante la GVL) se halla implementada y lista para usar. Además, contar con la colección de recursos que una librería de este tipo supone, permite enriquecer las visualizaciones (de grafos en este caso) con las facilidades provistas. Además, dado el hecho de que VTK es una librería extensible y en constante evolución, utilizarla como base para el desarrollo permite que la incorporación de nuevas características a futuro se realice de manera simple contribuyendo a la extensibilidad y mantenibilidad de la aplicación

Limitaciones

A continuación se detallarán las principales limitaciones para la visualización interactiva de grafos existentes en el prototipo, algunas de las cuales provienen del uso de la GVL y VTK y otras de la implementación actual. Dichas limitaciones son:

- El formato de archivo gml soportado por la librería, en el estado actual de su implementación en la GVL no se adapta completamente al estándar GML. Además el hecho de que sea el único formato soportado limita el abanico de ejemplos que se pueden utilizar para evaluar este prototipo. Sin embargo esta última restricción podría evitarse implementando conversores o generadores de grafos manualmente.

- El lector de archivos gml realiza una lectura completa del grafo. Esto condiciona los tiempos de carga indefectiblemente para grafos grandes. Una alternativa sería el permitir la obtención de determinado subconjunto de nodos/arcos desde el archivo. De esta manera se podría proporcionar al usuario una vista parcial para que inicie la exploración del grafo sin la necesidad de cargarlo todo en memoria. Los nodos no cargados inicialmente se irían obteniendo bajo demanda. Sin embargo el hecho de leer nodos/arcos desde archivo según la demanda del usuario puede comprometer los tiempos de respuesta de la aplicación y comprometer su interactividad.

- En la Graph Visualization Library [GVLVtk], los filtrados se realizan en base a métricas. Si bien esto no es malo en sí mismo, el hecho de utilizar métricas implica que deban calcularse en todos los nodos de cada grafo que se desea visualizar. El tiempo de cómputo de dichas métricas en grafos muy grandes se torna apreciable al usuario. Aunque ciertas métricas no se recalculan periódicamente (pueden implementarse para que se calculen al cargar el grafo) aumentan considerablemente los tiempos de carga. Además aquellas métricas que sí se recalculan periódicamente (por ejemplo la métrica basada en la localidad del nodo seleccionado) pueden comprometer los tiempos de respuesta de la aplicación y de esta manera degradar la interactividad de la aplicación con el usuario.

- El algoritmo Gem de layout basado en fuerzas provisto por la Graph Visualization Library [GVLVtk] se calcula hasta la estabilización completa del sistema. Esto redunda en tiempos de ejecución inaceptables aún para grafos no tan grandes.

- Al calcular el layout luego de la etapa de filtrado el sustrato espacial aportado por dicho algoritmo de layout se calcula en base a un subconjunto de los nodos del grafo. Esta es una limitación fuerte que se hace evidente durante los filtrados. Si consideramos el caso (soportado por la herramienta) en el cual el usuario filtra los nodos que se encuentran a una distancia mayor a determinado valor respecto al nodo seleccionado, el usuario logra visualizar al nodo seleccionado y aquellos nodos cercanos al mismo. Cuando este selecciona un nodo diferente, para ser considerado actual, se recomputa la métrica y se filtra obteniendo un nuevo subconjunto de nodos al cuál aplicar el algoritmo de layout. Aplicar el layout en estas condiciones resulta en que el layout calculado no tiene ninguna relación con el anterior (aún aplicando el mismo algoritmo). Esto representa una gran dificultad para el usuario ya que pierde el mapa mental que se había construido a partir de la vista anterior. Para salvar esta dificultad la Graph Visualization Library [GVLVtk] provee un filtro que permite aplicar el layout de un vtkGraph en otro. Este problema se soluciona entonces, calculando el layout para todo el grafo del archivo gml y almacenándolo en un vtkGraph que no será visualizado. Luego se filtra normalmente el grafo y se aplica el layout que se almacenó. Todos los nodos del grafo filtrado toman las posiciones en el espacio precalculadas. De esta manera si un nodo se visualiza en una secuencia de vistas filtradas, se ubicará exactamente en la misma posición. El problema en este caso es que debe calcularse siempre el layout para todo el grafo lo cual involucra un gran costo. Además si consideramos que uno de los motivos interesantes para filtrar es el hecho de aumentar la performance de la herramienta (sobre todo con grafos muy grandes), esta aproximación estaría anulando este beneficio. Una solución definitiva al respecto consistiría en utilizar técnicas de layout incrementales en el sentido que permitan agregar nuevos nodos a un conjunto de los mismos calculando el sustrato espacial sólo para los nuevos nodos y sin modificar la posición de los nodos que se visualizaban con anterioridad. Los algoritmos basados en fuerzas que incorporan animación son un ejemplo de estas técnicas de layout.

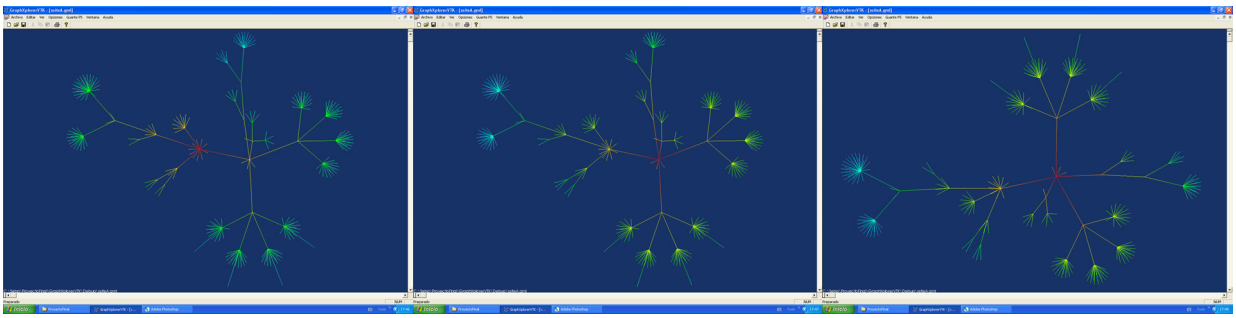

Fig. 4: En la secuencia de imágenes se aprecia el problema que existe con el mapa mental del usuario en un algoritmo de layout. La imagen de la izquierda es un grafo cuyo nodo actual se halla coloreado de rojo. Al preservar el mapa mental y seleccionar como actual el nodo vecino, el layout se conserva (imagen central). Si por el contrario no se preserva el mapa mental, la selección de un nuevo nodo actual involucra recalcular el layout con el consiguiente reposicionamiento de los nodos y arcos en el grafo (imagen derecha).

Interacciones sobre grafos

El prototipo cuenta con algunas interacciones básicas propias del área de Visualización de Grafos. Es posible modificar el punto de vista y navegar el espacio de representación y además seleccionar un nodo como el actual. También permite aplicar métricas, filtrados, algoritmos de layout, seleccionar elementos y atributos visuales y todas las cuestiones que se describieron cuando se detalló el pipeline del prototipo.

En el prototipo se han implementado algunas interacciones que se consideraron útiles a la hora de visualizar grafos. Dado el uso de vistas en 3D, las posibilidades de navegación de dicho espacio tridimensional son básicas. La aplicación permite, basándose en una metáfora de cámara en el espacio, realizar operaciones típicas como son rotaciones, traslaciones, zoom geométrico y otras transformaciones sobre la cámara.

La noción de un nodo actual a partir del cual conducir la navegación del usuario en el espacio de información se modeló en la aplicación. Se permite la posibilidad de seleccionar el nodo actual desde la vista y utilizar dicha información para realizar filtrados en base a distancia que permitan discriminar el foco de atención del usuario del contexto de su exploración.

Si bien se reconoce la utilidad de expandir / colapsar los vecinos de un nodo o filtrar determinado nodo directamente desde la vista, este tipo de interacciones aún no se halla soportada en el prototipo.

En relación a la gran cardinalidad de los grafos en determinados dominios, la aplicación permite filtrar. Se ha identificado el clustering de grafos como una interacción necesaria.

Conclusiones y trabajo futuro

El presente trabajo permitió mostrar la naturalidad con la que se enmarca la Visualización de Grafos en el Modelo Unificado de Visualización [MCFE03] desde el punto de vista del implementador de una visualización para un dominio de aplicación. Esto constituye la base para determinar los operadores y operandos principales de este dominio para posibilitar la validación completa del mencionado modelo. Se ha relevado un conjunto de operandos y operadores inicial que permitirá luego evolucionar al conjunto de operandos y operadores definitivos del MUV. Esto es esencial para completar un diseño de prototipo que pueda ser testeado por un usuario de la visualización y validar así el MUV desde el punto de vista del mismo.

En este sentido se trabajará en distintos aspectos como:

- Refinar los mecanismos de interacción con el usuario para hacer más natural el uso de la herramienta. Se evalúa la posibilidad de probar dispositivos no convencionales para interactuar con las vistas de la aplicación.

- Incorporar la animación de los algoritmos de layout en el modelo de ejecución de VTK [VTK] para sortear, por ejemplo, las limitaciones que surgen cuando se intenta preservar el mapa mental del usuario.

- Tener en cuenta escalabilidad realizando clustering de grafos tomando como punto de partida la métrica Strahler provista por la Graph Visualization Library [GVLVtk]. Dicha métrica da una indicación de la complejidad estructural del grafo alcanzable desde cada nodo y puede utilizarse para estos fines.

Sin duda el enfoque sobre una aplicación particular ayudará a cubrir la distancia entre el modelo teórica y las aplicaciones del mismo y es un modo de establecer el valor de la propuesta realizada.

Adicionalmente, el trabajo realizado permitió obtener una visión tanto de las facilidades que nos provee una librería como la Graph Visualization Toolkit [GVLVtk], como de sus limitaciones. Además nos permitió evaluar la posibilidad de construir una aplicación sobre el Pipeline de Visualización de VTK [VTK].

A lo largo del desarrollo, han surgido una innumerable cantidad de aspectos muy valiosos dado que ilustran las problemáticas del área desde el punto de vista del implementador. De la construcción y el uso del prototipo se obtienen conclusiones y mejoras impensadas al momento de diseñar la aplicación o plantear el modelo a usar.

A lo largo de este trabajo se ha sumado experiencia y se han evaluado tecnologías de implementación con el propósito de encarar la evolución del prototipo propuesto. Como resultado del refinamiento iterativo que involucra la metodología adoptada, se irán evaluando características, a medida que se incorpora funcionalidad a la aplicación. De esta manera se aprovecha la realimentación originada a partir del uso que un esquema así supone y que contribuye a obtener resultados acordes a las necesidades del usuario en menor tiempo.

En el corto plazo continuará la evaluación de la Graph Visualization Library [GVLVtk] explorando las posibilidades que provee para el manejo de atributos asociados a nodos y arcos. Si bien VTK [VTK] permite asociar valores numéricos a los nodos (por ejemplo sucede eso cuando se computan las métricas) dicha librería proporciona facilidades para asociar información textual (cadenas de caracteres) a los elementos de un grafo.

También y en relación con lo anterior se evaluarán ciertos filtros de la Graph Visualization Library [GVLVtk] que permiten realizar operaciones como copias, uniones, intersecciones y diferencias de grafos. Esto permitiría implementar en el prototipo facilidades para que el usuario modifique los filtrados directamente a partir de la vista. Por ejemplo si el usuario desea expandir un nodo (posiblemente distinto del actual) para visualizar sus vecinos (o su contexto), dicha operación podría implementarse como la unión del grafo actualmente visualizado con un grafo que resulta de filtrar (según la métrica de localidad y el nodo elegido por el usuario) el grafo completo. Con un esquema similar y la operación de diferencia se podría ocultar nodos según la demanda del usuario.

Referencias

[CMS99] S. Card, J. Mackinlay, B. Shneiderman, *Readings in Information Visualization - Using Vision to Think*, Morgan Kaufmann, 1999.

[DETT99] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice Hall, 1999.

[GVLVtk] Graph Visualization Library - http://www.cs.bath.ac.uk/~djd/graphs.html

[HMM00] I. Herman, G. Melançon, S. Marshall, *Graph Visualization and Navigation in Information Visualization: a Survey*, IEEE Transactions on Visualization and Computer Graphics, Vol. 6, No. X, XXX, 2000.

[MCFE03] S. Martig, S. Castro, P. Fillottrani, E. Estévez, *Un Modelo Unificado de Visualización*, Proceedings, pp. 881-892, 9º Congreso Argentino de Ciencias de la Computación. 6 al 10 de Octubre de 2003. La Plata. Argentina.

[VTK] VTK – The Visualization Toolkit – www.vtk.org. Kitware – www.kitware.com.