

Ka-Map!



Una moderna interfaccia per il Web Mapping che sfrutta AJAX ed il Mapserver

di Lorenzo Becchi e Andrea Cappugi

Abstract

Se il sito web Google Maps e la sua versione desktop Google Earth sono ormai sulla bocca di molti è grazie alle loro grande interattività e all'innovativa capacità di rendere la navigazione delle mappe un'esperienza semplice e accessibile. Il mondo del Free Software non è rimasto a guardare e già dal maggio 2005 la canadese DM Solutions ha rilasciato con licenza Open Source la sua ultima interfaccia per mapserver: **ka-Map**

Ka-Map raccoglie la sfida di Google Maps e trova il modo di rilanciare con la consueta duttilità dei software Open Source. La comunità ha risposto con un forte impulso di sviluppo e adesso ka-Map è la soluzione open source più apprezzata per visualizzare mappe sul web.

Perché ka-Map

UMN Mapserver è l'applicazione di riferimento per ciò che riguarda il Web Mapping e i Web Services. E' un dato di fatto che questa robusta applicazione stia rosicchiando quote di mercato ai ben più blasonati software commerciali ed il motivo va trovato nella sua estrema robustezza e nella grande comunità di sviluppo che continua a sostenere il progetto. Mapserver offre una interfaccia di default che è decisamente basilare ma molti progetti si sono incaricati di fornire integrazioni e soluzioni alternative capaci di soddisfare le più disparate richieste. Si va da interfacce essenziali a strumenti molto simili ad applicazioni GIS.

Ognuna di loro ha delle caratteristiche comuni ma soprattutto si porta dietro i limiti delle interfacce web classiche:

- **Impossibilità di effettuare panning in modo fluido;**
 - Le operazioni di panning, come quelle di zoom, richiedono il completo caricamento della pagina per ogni movimento. Questo comporta una esperienza di navigazione alquanto frammentata.
- **Estrema lentezza;**

- la necessità di scaricare continuamente l'intera pagina web per ogni interazione richiesta porta ad un notevole traffico dati non necessario con conseguente allungamento delle attese per raggiungere l'informazione desiderata.
- **Interfaccia utente generalmente poco intuitive**
 - L'utente non esperto di software GIS si trova a dover compiere una serie di operazioni prima di poter raggiungere

Una nota a parte la meritano le interfacce con applet Java. Queste soluzioni offrono una esperienza molto più simile a quella delle applicazioni desktop ma hanno il problema di essere pesanti da scaricare e spesso lente nell'interagire ai comandi dell'utente.

In questo contesto era palese la necessità di ricorrere a nuove tecnologie per offrire un'esperienza di navigazione più naturale. Google ha offerto la sua soluzione Google Maps mentre la comunità Open Source ha risposto con ka-Map.

ka-Map

ka-Map è un **progetto open source** creato per offrire una interfaccia **Ajax** per sviluppare applicazioni di web-mapping altamente interattive sfruttando le funzionalità disponibili nei moderni browsers.

Cos'è AJAX?

Dall'articolo di **Jesse James Garrett**

(<http://www.adaptivepath.com/publications/essays/archives/000385.php>):

Non vi è dubbio che la diffusione di internet e il miglioramento costante della velocità di connessione hanno permesso la creazione di numerose applicazioni Web. Quasi tutti gli utenti internet hanno avuto esperienze di utilizzo di almeno un programma di posta elettronica che avesse una interfaccia Web (es: Yahoo Mail). Nonostante la grande diffusione di queste applicazioni non è difficile notare che le loro interfacce perdano qualcosa in confronto alle applicazioni desktop (es: Outlook Express). La stessa semplicità che ha permesso la rapida proliferazione del Web ha anche creato il distacco fra l'esperienza offerta dalle applicazioni Web e quella delle applicazioni Desktop.

Adesso questo distacco si sta chiudendo. Provate ad usare Google Suggest (<http://suggest.google.com>). Mentre scrivete nel campo di ricerca quasi istantaneamente arrivano i suggerimenti. Adesso visitate Google Maps (<http://maps.google.com>). Provate a fare un ingrandimento (zoom in). Adesso portare il cursore del mouse sulla mappa e, mantenendo il pulsante premuto, trascinate la mappa nelle varie direzioni. Tutto avviene quasi istantaneamente senza l'attesa per ricaricare la pagina web.

Google Suggest e Google Maps sono quindi due ottimi esempi di questo nuovo approccio alla programmazione delle applicazioni Web che viene chiamato Ajax. Si tratta dell'acronimo di Asynchronous JavaScript e XML.

Definizione di Ajax

Ajax non è una tecnologia. E' in realtà più tecnologie indipendenti l'una dall'altra ma unite in un nuova e potente maniera. Ajax offre:

- Presentazione basate sugli standard di XHTML e CSS;
- visualizzazioni dinamiche ed interazioni basate sul Document Object Model (DOM);
- scambio e manipolazione di dati utilizzando XMLHttpRequest;
- Javascript come piattaforma per unire e gestire le suddette tecnologie.

Il modello classico delle applicazioni web funziona così: la maggior parte delle azioni nell'interfaccia lancia al web server una richiesta HTTP. Il server processa la richiesta (registrando dati, processando numeri, comunicando con sistemi distribuiti) e quindi rende al client una pagina HTML. E' un modello che è nato con il Web come mezzo ipertestuale.

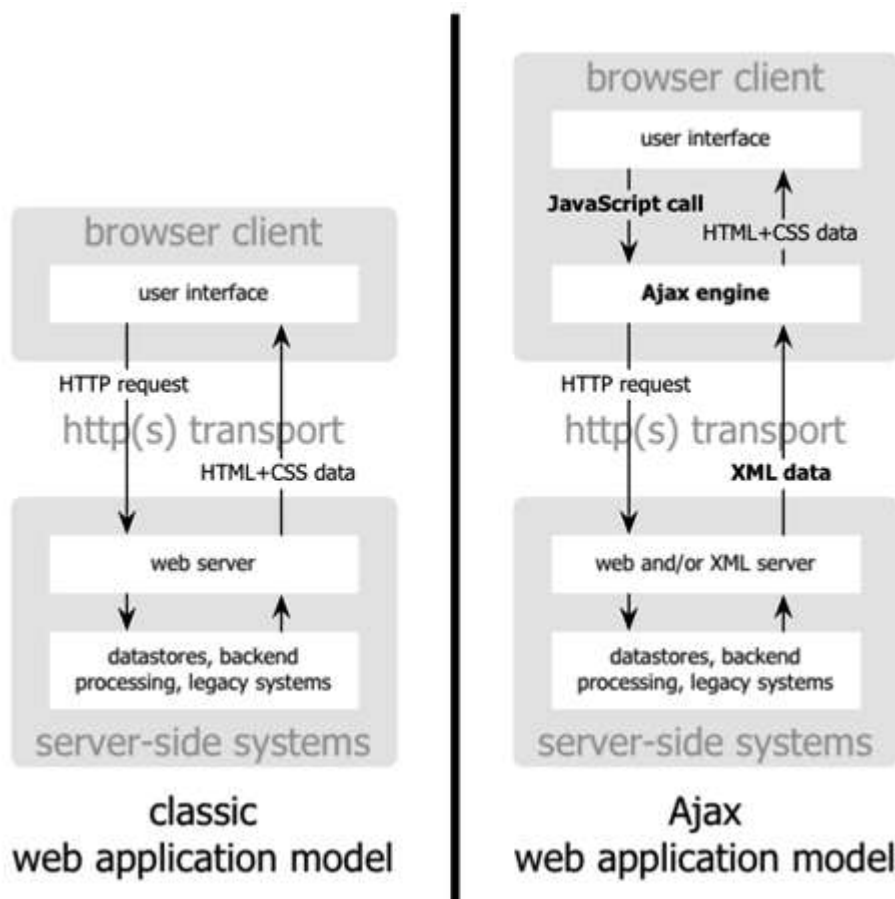


Fig 1: confronto procedurale fra il modello Web classico e modello Ajax

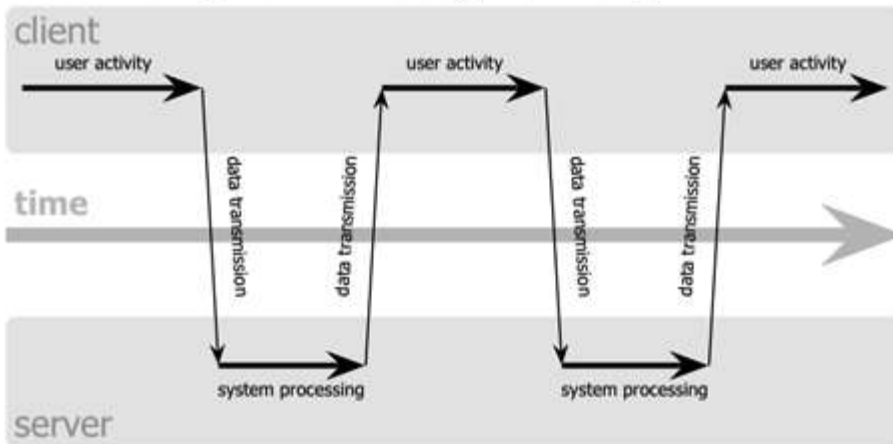
Fonte: <http://www.adaptivepath.com/publications/essays/archives/000385.php>

Il sistema è davvero sensato ed efficiente ma questo non porta ad una altrettanto sensata esperienza per l'utente. Mentre il server fa le sue cose, cosa ci si aspetta che faccia l'utente? ogni passo del processo porta l'utente ad aspettare ancora un po'.

Perché Ajax è differente

Una applicazione Ajax elimina la natura intermittente dell'interazione utente-server inserendo un intermediario: un motore AJAX. Invece di caricare pagine web dall'inizio, il browser scarica il motore Ajax scritto in javascript e generalmente invisibile. Questo motore è responsabile di creare l'interfaccia che l'utente vede e, allo stesso tempo, di comunicare con il server in base alle azioni dell'utente. L'interfaccia Ajax permette di rendere asincrona l'interazione fra utente e server. In questo modo l'utente non si trova mai di fronte a pagine bianche in attesa che il server faccia qualcosa.

classic web application model (synchronous)



Ajax web application model (asynchronous)

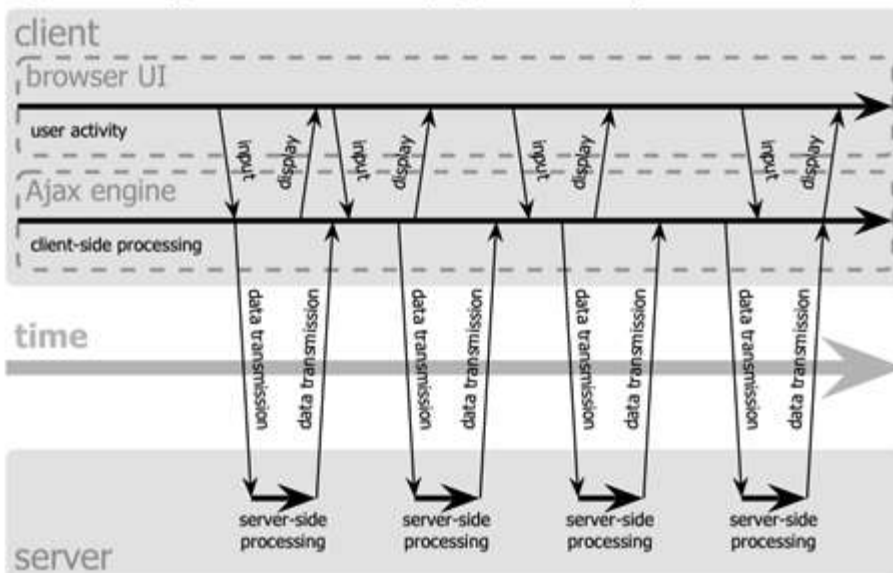


Fig 2: confronto operativo fra il modello Web classico e modello Ajax

Fonte: <http://www.adaptivepath.com/publications/essays/archives/000385.php>

Ogni azione dell'utente che normalmente finirebbe per generare un richiesta HTTP prende forma di una richiesta JavaScript all'interfaccia Ajax. Ogni risposta verso le azioni dell'utente che non necessitano di connessioni al server (come validazione dei dati, editing dei dati in memoria e anche qualche azione di navigazione) sono gestite direttamente dall'interfaccia. Se si presenta la necessità di ricorrere al server (invio di dati, caricare nuove parti dell'interfaccia o richiesta di nuovi dati) le connessioni al server avvengono un maniera asincrona, usando normalmente XML, senza interrompere l'interazione fra l'utente e l'applicazione.

Caching system

Caratteristica fondamentale comune a Google Maps e ka-Map è il sistema di caching.

Le applicazioni standard di Mapserver non utilizzano questo sistema ed ogni volta che una mappa viene richiesta, anche solo per uno spostamento o cambiamento di scala, il Mapserver deve accedere ai dati e ricreare le immagini da rendere al client. Questo comporta che il web server passa la richiesta al Mapserver, il Mapserver ricerca i dati dalle varie fonti, elabora la risposta e rende l'immagine al web server ed infine questa immagine viene inviata, con tutta la pagina web, indietro al client dell'utente.

In ka-Map i dati raccolti dal Mapserver sono trasformati in immagini in formato di tessere (tiles) di 200 px di lato. Ogni volta che viene visualizzata una certa mappa ad una certa scala, ka-Map si incarica di controllare che le tessere vengano create solo per l'area richiesta. Se queste tessere esistono già, questo è il punto importante, le tessere non vengono ricreate ed il server web restituisce direttamente le immagini senza dover consultare il Mapserver. Così si risparmiano notevoli risorse di calcolo e quindi tempo nella risposta.

In pratica ka-map salva in cache solo le aree richieste dagli utenti, questo determina una strategia mista per quanto riguarda i problemi: ridondanza dei dati - velocità dell'applicazione. Sono gli utenti stessi che determinano cosa ed a che scala verrà salvato in cache evitando un'eccessiva ridondanza dei dati.

Per chi richiede le massime prestazioni in velocità, esiste inoltre uno script (precache.php) che si incarica di processare una volta per tutte le tessere di una certa mappa a tutte le scale previste dalla sua configurazione. Questo permette a chi sviluppa una applicazione con ka-Map di pubblicare un sito solo quando le tessere siano già state create, permettendo ai visitatori di avere da subito la migliore esperienza di navigazione possibile. La velocità di risposta dell'applicazione è quindi relazionata alla velocità di connessione disponibile per l'utente, alla velocità di connessione disponibile per il server e al tempo di accesso alle tessere sul disco. E' facile notare che le variabili limitanti sono ridotte al minimo.

Configurazione flessibile basata su UMN MapServer

Come abbiamo precedentemente descritto, ka-Map svolge la funzione di interfaccia grafica lato client per Mapserver. La gestione delle connessioni ai dati, l'elaborazione e la preparazione delle immagini viene svolta interamente dal Mapserver. Gran parte della personalizzazione per le applicazioni realizzate con ka-Map dipende dal mapfile che non è altro che il file di configurazione che il Mapserver utilizza per definire il contorno (extent, scala di inizio, proiezione, formato di output, ecc.) ed i singoli livelli di dati (nome, stato, tipo, sorgente dati, scala minima, scala massima, ecc.) della mappa da produrre.

La parte di configurazione che resta al ka-Map riguarda la dimensione delle tessere, il numero di mapfile da presentare, con che nome presentarli e quali livelli di zoom offrire.

Naturalmente qualche programmatore più smaliziato potrebbe mettere le mani dentro il

codice e personalizzare anche l'interfaccia aggiungendo nuove funzionalità, modificando le esistenti o cambiando l'aspetto grafico a suo piacimento. Per fare questo è consigliata una certa dimestichezza nello sviluppo di applicazioni AJAX.

Mappe e layer (group) multipli

Abbiamo appena detto che la personalizzazione di ka-Map permette di presentare più mappe con la stessa applicazione. Le mappe possono essere totalmente diverse le una delle altre in quanto ognuna di loro fa riferimento ad un distinto mapfile.

Per ciò che riguarda la gestione dei livelli (LAYER), ka-Map ha una sua peculiarità. Sfrutta l'attributo GROUP definibile per ogni LAYER del mapfile per creare quelli che sono i veri e proprio livelli dell'applicazione. In questo modo più LAYERS possono appartenere ad uno stesso gruppo e ka-Map li rappresenterà raggruppati su uno stesso livello, unendo in una singola immagine tutti i layers appartenenti al medesimo gruppo. Quindi la legenda di ka-Map presenterà per un livello per ogni GROUP e nella rispettiva legenda saranno presenti i simboli per ognuno dei LAYER appartenenti al dato GROUP. Le proprietà dei vari livelli, descritte in seguito, verranno applicati a tutti i LAYER del GROUP relativo.

Navigazione da tastiera

Per migliorare la navigabilità, ka-Map permette di utilizzare alcuni comandi da tastiera per spostarsi sulla mappa nelle varie direzioni.

Ipotizzando di aver utilizzato un sistema di proiezione classico, mappa orientata con il nord in alto e ovest a sinistra, il tasto **up** permette di spostare la centratura della mappa verso Nord, il tasto **down** verso sud, **left** verso ovest, **right** verso est. Allo stesso modo si possono usare i tasti **page up**, **page down**, **home**, **end** per spostarsi nelle rispettive direzioni per una distanza maggiore.

funzionalità estensibili attraverso una tool API

Le funzionalità base coprono già numerose necessità ma chi non sapebbe accontentarsi potrebbe utilizzare le **API** (Application Programming Interface) a disposizione per estendere le potenzialità di ka-Map.

Questa opzione richiede un discreto livello di dimestichezza con la programmazione in ambiente AJAX ma l'accesso non è proibitivo. La documentazione a riguardo non è ancora buona ma le funzionalità sono documentate nel codice e il WIKI (<http://ka-map.ominiverdi.org>) della comunità di ka-Map offre già numerosi chiarimenti.

Interfaccia Utente

ka-Map, facendo proprie le caratteristiche di molti software open source, si presta a differenti implementazioni della sua interfaccia. Tra le principali troviamo quella di default e quella embedded.

Default

Come è facile notare dall'immagine allegata, l'implementazione di default si presenta come una normale applicazione desktop per la navigazione delle mappe. Nell'intestazione (Toolbar) sono presenti tutti i pulsanti per accedere alle funzioni di Zoom In, Zoom Out, Zoom a scala predeterminata, Zoom alla scala di default. Inoltre è presente la select box per scegliere fra le varie mappe definite nell'implementazione. Altri due pulsanti permettono di scegliere fra la modalità di Panning (scorrimento) e quella di Identify (con click singolo o rubber box).

In basso a destra rispetto alla Toolbar è presente la mappa di riferimento (Keymap). Nella mappa di riferimento vi è il selettore della visione corrente che può essere utilizzato per cambiare vista sia tramite il doppio click in un punto della Keymap sia trascinando il rettangolo del selettore (riquadro con bordo rosso).

Sotto la Keymap vi è una scalebar (Scalebar) che funziona integralmente lato utente, ovvero non viene richiesta l'immagine al server tutte le volte che si cambia scala.

Sotto la Scalebar si trova la legenda (Legend). Questa è la parte più interattiva di tutta l'interfaccia e presenta numerose funzioni. I pulsanti "+" e "-" al lato di ogni livello permettono l'espansione e la chiusura delle singole legende. Per ogni livello è possibile attivare o disattivare la visibilità agendo su un checkbox, regolare la opacità delle immagini, cambiare la posizione in alto o in basso.

Accanto a Toolbar, Keymap e Scalebar è presente una freccia che permette di far apparire e scomparire i riquadri a piacimento. Nascondendo i riquadri si ottiene una navigazione a tutta pagina che nessun altro sistema tradizionale può offrire.



Fig 3: *Interfaccia Standard*

Fonte: <http://www.ominiverdi.org/>

Embedded

L'utilizzo del tag HTML IFRAME permette di inserire in qualsiasi pagina web un riquadro contenente una mappa interattiva che può essere caricata da qualsiasi server remoto.

Tra gli attributi di IFRAME vi è appunto SRC che permette di definire la sorgente dei dati da mostrare. tra le interfacce di ka-Map vi è quella che fa riferimento al file `iframe.html` che offre una soluzione semplificata senza Keymap, Scalebar e Legend e con una Toolbar minimale.



Fig 4: *Interfaccia Embedded*
Fonte: <http://www.ominiverdi.org/>

Compatibilità

Allo stato attuale la compatibilità è al livello delle più blasonate soluzioni AJAX (Google Maps, ecc.).

- Internet Explorer 6.0 o superiore
- Firefox 0.8 o superiore
- Safari 1.2.4 o superiore
- Netscape 7.1 o superiore
- Mozilla 1.4 o superiore
- Opera 8.02 o superiore

come funziona ka-Map

Abbiamo già accennato alla gestione delle tile o tessere che dir si voglia. La gestione di queste a livello del client avviene attraverso due elementi fondamentali: **theInsideLayer** e il **viewport**. Le tessere sono posizionate nel 'pixel space' con coordinate assolute.

theInsideLayer

Il layer interno, **theInsideLayer** è la struttura dove si crea la mappa. In esso il motore Ajax si incarica di applicare le varie tessere che vengono mano a mano scaricate dal server. Nella griglia del theInsideLayer si sviluppa l'immagine della mappa così come le vediamo durante la navigazione.

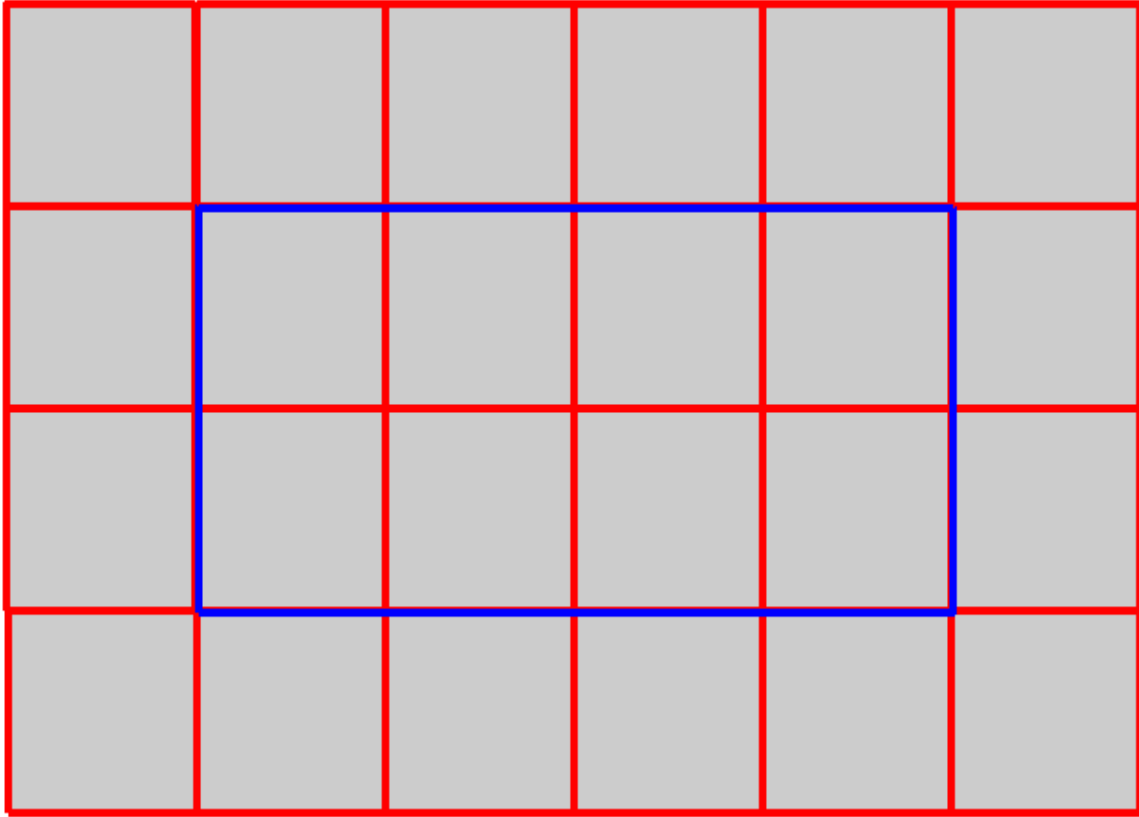


Fig 5: struttura dell'InsideLayer
fonte: *Paul Spencer, DM Solutions Group, 2005*

viewport

Il **viewport** è una finestra che permette all'utente di visualizzare una parte del contenuto dell'insideLayer.

Nell'azione del panning (scorrimento della mappa) theInsideLayer scorre sotto il viewport. E' il viewport che intercetta le azione di movimento e manda le richieste asincrone per ottenere il download delle tessere di mappa che lo circondano.

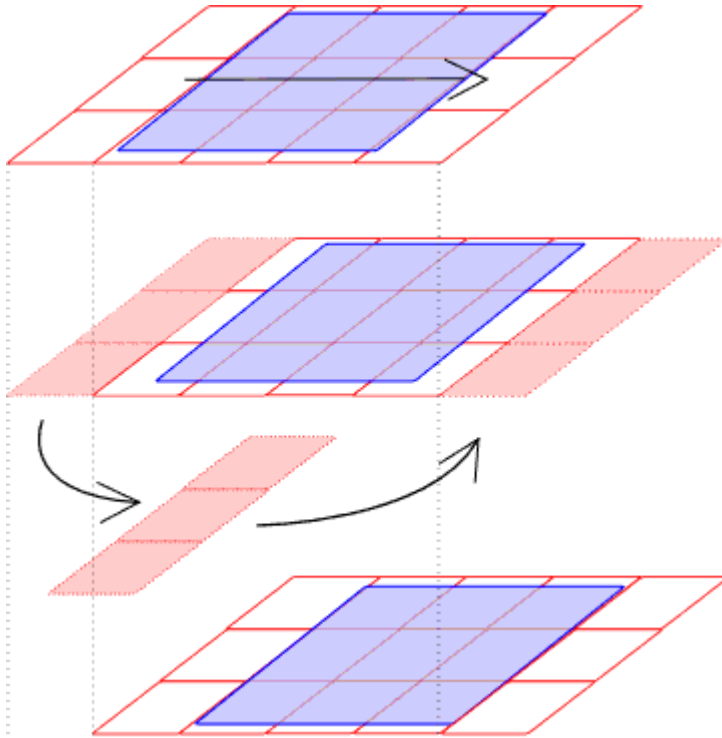


Fig 6: procedura di caching durante il panning.

fonte: Paul Spencer, DM Solutions Group, 2005

Conclusioni

Ka-Map è un'applicazione giovane e come tale non può avvalersi di un curriculum di grande prestigio. Siamo ancora alla versione **0.2**. Se lo sviluppo gode di un'ottima salute e nel CVS del progetto si possono trovare numerose nuove funzioni. L'interesse della comunità così come le sponsorizzazioni permettono ai software open source di crescere o scomparire. Ka-Map compie il suo primo anno dimostrandosi uno progetto in ottima salute. L'anno che verrà mostrerà tutto il valore di questa applicazione.

Questo software offre l'occasione a molti siti che offrono web mapping di rinnovare il loro aspetto e le loro potenzialità. I limiti alle implementazioni di questa applicazione riguardano solo la fantasia degli sviluppatori e la disponibilità di dati.

Non perdetevi i Workshop e le presentazioni che verranno fatte a Losanna il prossimo settembre (<http://www.foss4g2006.org>). Sarà l'occasione per tastare il polso al crescente mondo del Free and Open Source Software for GIS. Ka-Map ci sarà!

Link utili

- La Home Page di ka-Map: <http://ka-map.maptools.org/>

- il Wiki per la documentazione di ka-Map: <http://ka-map.ominiverdi.org/>
- la Home Page di Ominiverdi, co-sviluppatori di ka-Map:
<http://www.ominiverdi.org>
- Google Maps: <http://maps.google.com>