# Visualizzazione dei palinsesti televisivi

## Analisi dei requisiti

Spesso vi è la necessità di conoscere il palinsesto di uno (o più) canali televisivi senza la *fatica* di dover accendere la televisione e consultare il televideo, o la pagina del giornale contenente “*i programmi di oggi*”. Alcuni strumenti potrebbero essere quelli di utilizzare i siti Internet degli appositi canali (ad esempio: [Il palinsesto di Rai Uno](http://www.raiuno.rai.it/vgn-ext-templating/raiuno/index.jsp?vgnextoid=1bc5fde6d64a7010VgnVCM100000680212acRCRD)). Ma si consideri il processo: apertura del browser, clic sul bookmark o ricerca su Google del palinsesto del canale, consultare l’orologio e scegliere tutti i programmi il cui orario d’inizio è maggiore o uguale all’orario corrente. È chiaro che tale processo è un’attività noiosa e ripetitiva (consideriamo una persona che vuole conoscere ogni giorno i programmi televisivi della serata), e la perdita di tempo dovuta alla ricerca dell’informazione è improponibile (spesso i siti contengono banner, ads a tutto schermo, etc.). Senza contare che tale processo va ripetuto per tutti i canali di cui si vuole conoscere il palinsesto. Ci si potrebbe attrezzare con siti che raccolgono i palinsesti di più canali ([Il palinsesto di satellite.it](http://www.satellite.it/griglia.asp) è il miglior esempio): anche questi siti, però, sono caratterizzati da una lentezza di caricamento notevole (il palinsesto considerato utilizza addirittura la tecnologia Flash, che causa una lentezza considerevole nel caricamento e un’allocazione di memoria per il browser non indifferente), ricadendo quindi nel caso di uno spreco di tempo e risorse per ottenere una semplice informazione.

Frequentemente si sa che il proprio programma preferito sarà in onda in un determinato momento: basti pensare, ad esempio, alla frase “so che nei prossimi giorni ci sarà in onda il nuovo episodio di “Prison Break”, ma non so su che canale né a che ora…”: il processo di ricerca dell’informazione passa per la ricerca nei palinsesti dei canali del programma (in questo caso il film) desiderato. Come spiegato poco più sopra, tale processo è dispendioso per l’utente sia in termini di tempo che di risorse.

Infine, si ha poi un altro grande problema: spesso ci si dimentica che un programma che si vuole guardare sta per essere trasmesso in televisione; ad esempio: si pensi alla situazione in cui l’utente ha visionato il palinsesto futuro di un canale e ha deciso che il programma della serata è degno di nota. Se, durante la serata, l’utente *non si ricorda* di voler guardare tale programma, avrà perso definitivamente la trasmissione desiderata.

La soluzione del problema è *semplice*: visto che il processo è standard ed è sufficiente automatizzarlo, perché non avere un’applicazione che si occupi di fare il download dei palinsesti dei vari canali, consulti l’orologio (locale per l’utente) e mostri all’utente i programmi che sono in onda al momento? L’utente può anche ricercare dei programmi nel palinsesto, e selezionare un allarme per determinati programmi: in questo modo l’applicativo informa l’utente (tramite un reminder) che il programma sta per iniziare, fungendo da “agenda” dei programmi preferiti. In questo modo l’overhead di ricerca e selezione dell’informazione televisiva è commissionato all’applicazione, e l’utente può concentrarsi sull’attività “operativa” vera e propria: guardare i propri programmi di interesse

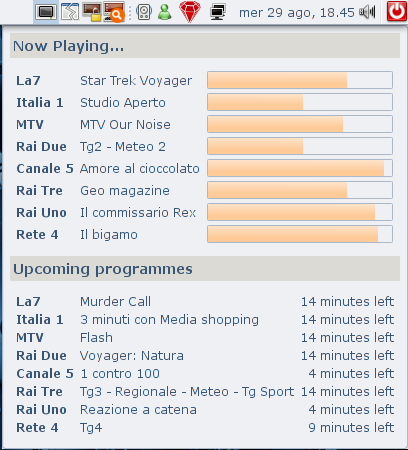
## Specifica dei requisiti funzionali

Si vuole realizzare un’applicazione che consenta di visualizzare il palinsesto di uno o più canali televisivi. In particolare, si vuole che l’applicazione sia dotata delle seguenti funzionalità:

* Specifica dei canali di cui si vuole conoscere il palinsesto (*canali di interesse*)
* Visualizzare i programmi in onda nel momento in cui viene interrogato. Inoltre, visualizzare i programmi che non sono ancora andati in onda
* Visualizzare il palinsesto di tutti i canali di interesse in modo continuo, utilizzando una rappresentazione che ne faciliti la comprensione (es. rappresentazione su un diagramma temporale).
* Ricerca, all’interno dei palinsesti, di programmi (*programmi favoriti*)
* Promemoria per ricordare l’inizio dei programmi favoriti

## Valutazione delle alternative

Attualmente esistono dei programmi che soddisfano alcune (ma non tutte) le necessità descritte nell’analisi dei requisiti: l’esempio più completo è sicuramente [OnTV](http://johan.svedberg.com/projects/coding/ontv/), un applet del desktop environment [Gnome](http://www.gnome.org). Tuttavia, la maggior parte di questi programmi è sviluppata per i sistemi operativi UNIX-Like.

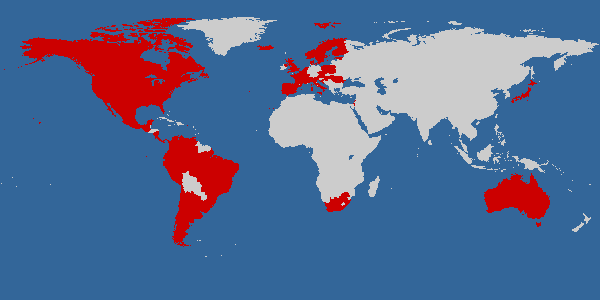


Inoltre, sempre in ambiente UNIX-like, si nota che *tutti* i programmi che consento di visualizzare il palinsesto si appoggiano ad un’applicativo esterno, [XMLTV](http://xmltv.org/wiki/). XMLTV è un programma a linea di comando che consente di disaccoppiare il problema di download dei palinsesti (backend) da quello della visualizzazione (frontend); infatti, la libreria è composta da una serie di grabber (esiste un grabber di palinsesti per ogni nazione, tra cui l’Italia). I grabber rispondono ad un file di configurazione che consente di selezionare i canali di interesse. Avendo selezionato i canali di interesse dal file di configurazione, si può lanciare il grabber per la specifica nazione: il grabber si occuperà del download dei palinsesti dei canali selezionati e produrrà un file XML con la descrizione dei palinsesti dei canali scelti.

Si è dovuto scegliere tra due alternative:

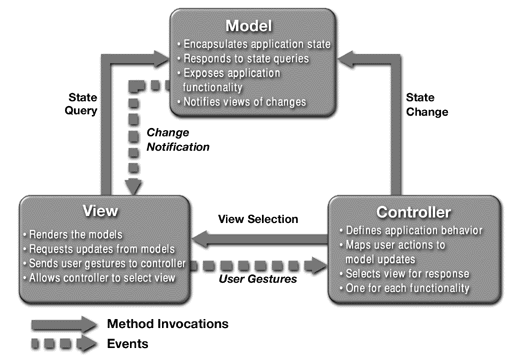
* Scrivere (da zero) un download-parser del palinsesto per ogni specifico canale e integrarlo nel programma. Nessuna separazione tra backend e frontend. Lo sforzo per lo sviluppo del download parser di un solo canale non è indifferente, e quindi il numero di canali televisivi integrabili nel programma sarebbe stato molto ridotto.
* Utilizzare XMLTV come programma esterno: il progetto costituisce il frontend di visualizzazione (parsing del file XML prodotto da XMLTV) dei programmi e integra un wrapper per utilizzare XMLTV. La lista dei canali è numerosa e aggiornata, l’algoritmo di download e di parsing è il migliore attualmente presente sul mercato. In questo modo si riusa del codice; infine, XMLTV è opensource.

Si è deciso di utilizzare l’idea di disaccoppiamento suggerita da XMLTV, sviluppando un’applicazione che contenesse un wrapper per il grabber di XMLTV. Il grabber è generico, in quanto XMLTV raccoglie un grabber per ogni nazione. In questo modo, l’applicazione sviluppata può appoggiarsi ad un grabber non italiano ed essere utilizzata anche da utenti non italiani! Infatti, è sufficiente che ci sia il grabber per la nazione selezionata: al momento esistono grabbers per: “Australia, Belgium and Luxembourg, Brazil, Argentina, Britain and Ireland, Croatia, Denmark, Estonia, Finland, France, Hungary and Romania, Iceland, Italy, Japan, Netherlands, North America, Norway, Portugal, Reunion Island (France), South Africa, Spain, Sweden, and Switzerland” – si veda [XmltvWorldDomination](http://xmltv.org/wiki/xmltvworlddomination.html)).



## Specifica dei requisiti non funzionali

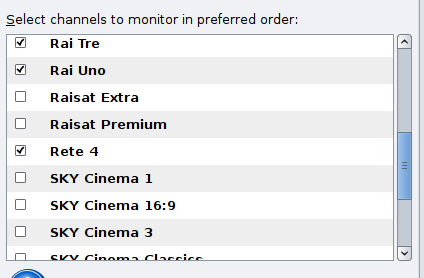
* Portabilità: il programma deve essere multipiattaforma: si utilizza, a tal scopo, il linguaggio di programmazione Java. XMLTV è opensource, e disponibile per le piattaforme UNIX-like e Windows
* Usabilità: il programma deve gestire XMLTV (tipicamente il grabber va lanciato dalla linea di comando) “nascondendo” all’utente la struttura backend-frontend: l’utente deve utilizzare l’interfaccia grafica solo per visualizzare e ricercare i palinsesti
* Efficienza: utilizzando la versione CVS di XMLTV, contenente le ultime migliorie ai vari grabber, il tempo di download di 3 giorni di programmazione per 6 canali televisivi non dovrebbe superare i 10 minuti
* Manutenibilità: il programma deve essere sviluppato secondo l’architettura MVC (Model-View-Controller) e deve garantire successive evoluzioni nel tempo



## Casi d’uso

### Use case #1 - opzioni

1. L’utente, nelle opzioni dell’applicativo, seleziona i canali di interesse (il programma seleziona la lista dei canali dal file di configurazione di XMLTV; l’applicativo scrive il file di configurazione di XMLTV, che contiene i canali scelti dall’utente) e il numero di giorni (in avanti) di cui vuol visionare il palinsesto



1. L’applicativo richiama XMLTV con il file di configurazione appena creato; una volta che XMLTV crea il file XML, l’applicativo ne fa il parsing, presentando all’utente il palinsesto dei programmi scelti

### Use case #2 - avvio

1. Quando l’applicativo viene avviato, controlla che non sia già presente il file XML con la descrizione del palinsesto
2. Se non è presente, l’applicativo schedula il download (attraverso XMLTV) di palinsesti aggiornati, sovrascrivendo i palinsesti attuali
3. Se è presente, controlla il periodo temporale di riferimento
4. Se il file *non è scaduto* (i palinsesti descritti contengono la data odierna ed eventualmente palinsesti più recenti)
5. Se il file è *scaduto* (i palinsesti descritti sono più vecchi rispetto alla data odierna), allora l’applicativo deve schedulare il download
6. L’applicativo mostra all’utente la programmazione televisiva

### Use case #3 – errore nel download

1. L’applicativo non riesce ad effettuare il download (attraverso XMLTV) di nuovi palinsesti (ad esempio perché manca la connessione ad Internet)
2. L’applicativo informa l’utente dell’errore

### Use case #4 – errore nel parsing

1. L’applicativo non riesce ad effettuare il parsing del file XML prodotto da XMLTV (ad esempio perché è corrotto)
2. L’applicativo informa l’utente dell’errore, e prova a fare il download (attraverso XMLTV) di nuovi palinsesti

### Use case #5 – ricerca

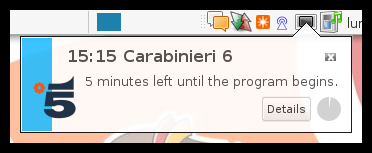
1. L’utente utilizza la funzione per ricercare un programma
2. L’applicativo ricerca la stringa alfanumerica inserita dall’utente all’interno dei palinsesti a disposizione (supponiamo, in questo caso, che siano aggiornati). Se uno (o più) programmi che corrispondono alla descrizione inserita, il programma comunica all’utente la data e ora di inizio, oltre al canale di messa in onda.

### Use case #6 – scelta di programmi preferiti e reminder

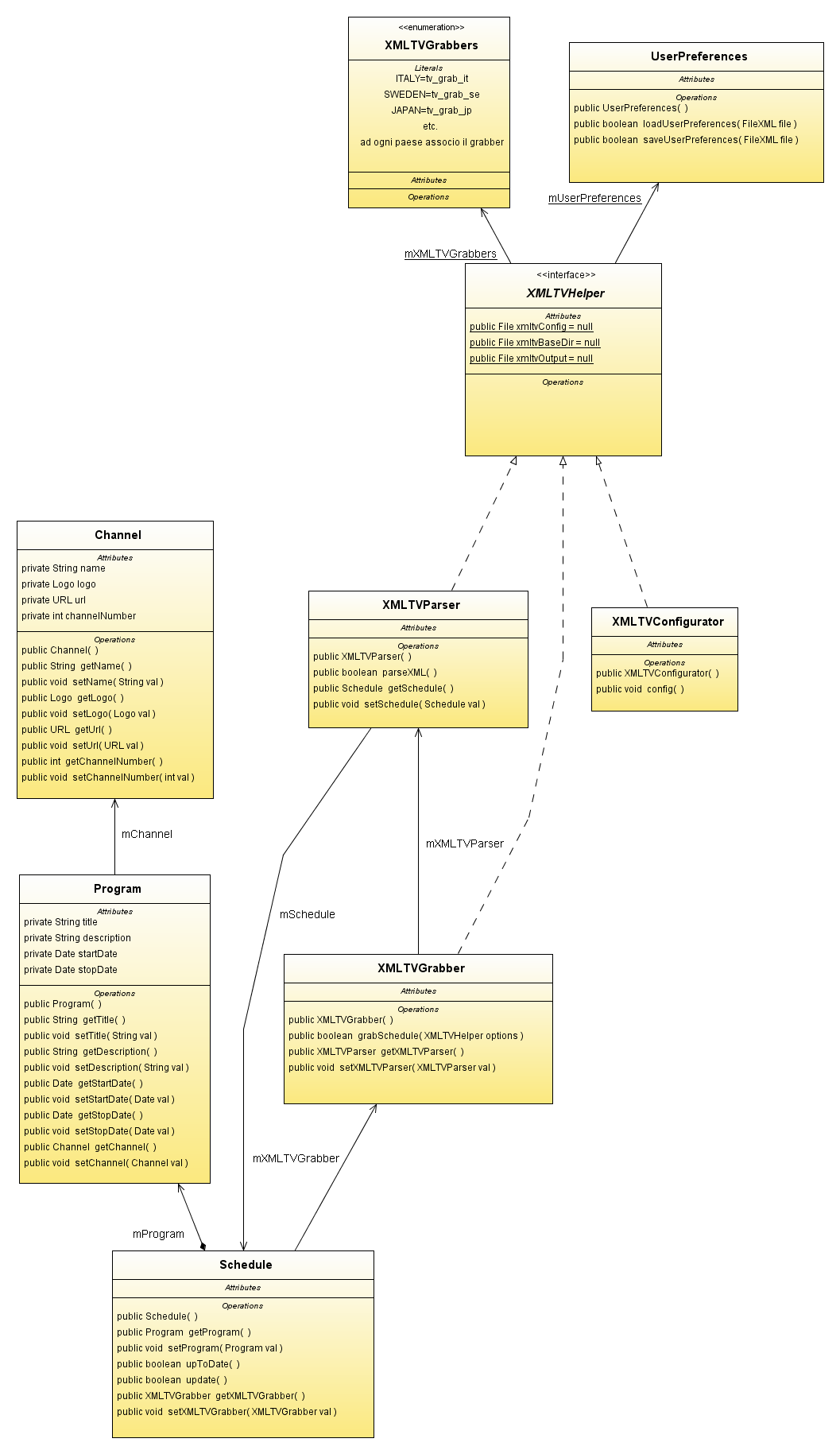
1. L’utente utilizza la funzione per ricercare un programma
2. L’applicativo ricerca la stringa alfanumerica inserita dall’utente all’interno dei palinsesti a disposizione (supponiamo, in questo caso, che siano aggiornati).
3. L’utente può selezionare i programmi trovati come preferiti
4. Il programma registra i programmi selezionati come preferiti, comunicando all’utente la data e ora di inizio, oltre al canale di messa in onda

### Use case #7 – reminder popup

1. Il programma si posiziona nel tray di sistema quando minimizzato
2. Quando mancano 10 minuti (opzione configurabile) all’inizio di un programma *preferito*, l’applicativo presenta un popup che avvisa l’utente



## Progettazione



## Funzionamento del programma

**MODEL**: modelliamo la tipologia dei dati che abbiamo a disposizione. Possiamo avere:

* Channel: descrive il generico canale di trasmissione con nome, id, logo, etc.
* Program: descrive il generico programma televisivo, ed è caratterizzato da orari di inizio e di fine, un titolo e una descrizione. Inoltre, contiene un riferimento (navigable association 🡪 mChannel) al canale su cui è trasmesso
* Schedule: rappresenta il palinsesto televisivo contenente tutti i programmi. È costituito da un insieme di canali (navigable composition 🡪 mProgram). Il palinsesto può essere aggiornato tramite XMLTV

Come si può capire se il palinsesto è aggiornato (metodo upToDate())? Descriviamo le operazioni in sequenza:

1. (controllo esistenza) Il file XML contenente l’output di XMLTV esiste? Se esiste, salta al 2. Se non esiste salta al 3.
2. (controllo upToDate()) Il file XML contiene programmi il cui inizio è la data di oggi (= palinsesto aggiornato)? Se è aggiornato, salta al 4. Se non è aggiornato, salta al 3.
3. Richiama il programma esterno XMLTV con le opzioni specificate dall’utente (contenute in UserPreferences)
4. Fine

Come funziona XMLTV e la relativa gestione?

Innanzitutto, XMLTV è un grabber di palinsesti. Ad ogni nazione corrisponde un grabber (ad esempio: per l’Italia si richiama tv\_grab\_it, per il Giappone tv\_grab\_jp). Per associare ogni nazione al grabber corretto, usiamo una classe Enumeration (XMLTVGrabbers). Introduciamo una classe interfaccia per maneggiare XMLTV: la chiamiamo XMLTVHelper. Per operare con XMLTV, è necessario specificare:

* Un file di configurazione, contenente i canali di cui si vuole scaricare il palinsesto (ad esempio specifico che voglio il palinsesto di Canale 5 ma non quello di Italia 1). Come si produce tale file? XMLTV viene distribuito con un file di configurazione già pronto per ogni grabber. Tale file contiene la lista di tutti i canali di cui è possibile scaricare il palinsesto mediante il grabber selezionato. Su ogni riga è presente un canale. Se la riga non è commentata, il grabber si preoccupa di scaricare il palinsesto di tale canale. La classe XMLTVConfigurator (implementation di XMLTVHelper) si preoccupa di scrivere (aiutando l’utente nella scelta) il file di configurazione.
* Un file XML di output, che conterrà il palinsesto scaricato da XMLTV
* La locazione (nel sistema) dell’eseguibile di XMLTV (su Linux, si deve chiamare direttamente il grabber, mentre su Windows si deve richiamare xmltv.exe seguito dal nome del grabber)
* Una serie di parametri (opzionali, ad esempio il numero di giorni di cui si vuole scaricare il palinsesto). Tali parametri sono contenuti nella classe UserPreferences, di cui XMLTV ha un riferimento (navigable association 🡪 mUserPreferences).

Per richiamare il programma esterno XMLTV, usiamo la classe XMLTVGrabber (implementation di XMLTVHelper): richiama il comando usando il file di configurazione e scrive l’output sul file specificato, utilizzando le opzioni eventualmente specificate dall’utente.

Una volta ottenuto il palinsesto, si deve fare il parsing XML del file prodotto da XMLTV. Questo è il compito della classe XMLTVParser (implementation di XMLTVHelper): fa il parsing del file XML e alloca le strutture dati descritte precedentemente: Channel, Program e Schedule.

Ritorniamo ora allo Schedule: quando un palinsesto non esiste oppure non è aggiornato, si deve richiamare XMLTV: lo Schedule contiene quindi un riferimento a XMLTVGrabber (navigable association 🡪 mXMLTVGrabber) per invocare XMLTV e portare a termine l’aggiornamento.

XMLTVGrabber contiene un riferimento a XMLTVParser (navigable association 🡪 mXMLTVParser) in modo da richiamare il parsing del palinsesto (file XML) scaricato da XMLTVGrabber.

Infine, XMLTVParser contiene un riferimento allo Schedule (navigable association 🡪 mXMLTVSchedule) in modo da riempire lo Schedule con i dati appena scaricati.

Si forma quindi il ciclo: Schedule 🡪 XMLTVGrabber 🡪 XMLTVParser 🡪 Schedule

Il ciclo non è di per sé un problema, ma non sono molto sicuro del flusso di dati tra questi 3 “attori”. Infatti, pensavo al metodo update() di Schedule: viene richiamato per avere un palinsesto aggiornato. Schedule sa che per avere tale palinsesto deve richiamare un grabber. Il grabber si occuperà di scaricare i dati e ritornare il palinsesto a Schedule, in modo trasparente per Schedule. Anche XMLTVGrabber richiama una classe esterna per fare il parsing dei dati appena scaricati: richiama infatti XMLTVParser. Ed è proprio XMLTVParser che andrà a riempire il palinsesto Schedule (infatti ha una navigable association 🡪 mSchedule).

Se il flusso di dati e chiamate descritto qui sopra fosse corretto, il dubbio che ho è che il parser debba avere un riferimento anche a Channel e Program.

**VIEW**: interfaccia grafica che mostra i dati dello Schedule in forma tabellare, mostrando per ogni Channel i programmi in onda.

**CONTROLLER**: mediante il mouse posso spostarmi tra la finestra principale contenente il palinsesto e le viste secondarie, come il configuratore (XMLTVConfigurator) e le opzioni (UserPreferences).