CMP Rai TV – Consuntivo Attività e Stato Avanzamento Lavori

La presente relazione fotografa lo stato attuale del progetto di estensione della **Consent Management Platform (CMP) di Rai** in ambito TV, con focus su **RaiPlay** al 25 Luglio 2025. Riporta le attività completate, il tempo-uomo investito e le stime economiche, sottolineando i risultati conseguiti, le scelte tecnologiche adottate e le opportunità previste per il prossimo step.

Attività concluse

Smart TV (Samsung Tizen, LG WebOS, browser web-based)

Il lavoro sull'ambiente Smart TV ha richiesto **445,5 ore di sviluppo**, pari a **55,7 giornate/uomo**. La stima economica dell'attività, calcolata secondo la tariffa standard di €500/giornata/uomo, è di **€27.850**. L'attività si è articolata in tre fasi principali: *Discovery*, *Sviluppo*, *Testing e Bugfixing*.

Dettaglio ore/uomo

Task: Banner su Samsung TV e altre Smart TV web-based: test di compatibilità per l'integrazione con lo script web, adattamenti UI CMP, integrazione navigazione tramite telecomando, fix per tecnologie non supportate.

Risorse: Fabio Politi, Cristiano Federico, Simone Mazzotti

Fase	Fabio Politi	Cristiano Federico	Simone Mazzotti	Totale Fase
Discovery	25,5	5,5	19,0	**50,0**
Sviluppo	105,5	25,5	55,0	**186,0**
Testing e Bugfix	115,5	28,5	65,5	**209,5**
Allineamenti e Call	17,0	10,0	23,0	**50,0**
Totale Persona	**263,5**	**69,5**	**162,5**	**445,5**

Lo stato attuale vede il banner correttamente integrato, funzionante e testato su un ampio parco dispositivi (Samsung Tizen OS, LG WebOS, Hisense VIDAA ecc.), con adattamenti mirati per ciascun ambiente e versione di sistema operativo. L'interfaccia è coerente con le linee guida grafiche fornite da Rai, rispetta le specifiche di progetto ed è completamente navigabile tramite telecomando.

Nota sugli effort di sviluppo

È importante sottolineare che gran parte dello sviluppo UI e della logica di navigazione realizzata per Smart TV è stata successivamente riutilizzata nell'implementazione Android TV, dove la CMP viene visualizzata all'interno di una WebView. Per questo motivo, e per praticità di rendicontazione, la maggior parte degli effort di sviluppo dell'interfaccia condivisa sono stati attribuiti al task Smart TV.

Problematiche riscontrate

Durante lo sviluppo sono emerse diverse criticità che hanno portato a un incremento significativo dell'effort, quadruplicando le stime iniziali. La complessità del progetto si è rivelata molto superiore alle aspettative già dalla fase iniziale:

- **Ridefinizione delle specifiche e compatibilità**: L'analisi iniziale prevedeva semplicemente il porting dello script esistente con adattamenti UI e implementazione della navigazione via telecomando. Tuttavia, già in fase di ri-definizione delle specifiche è emerso che la compatibilità richiesta si estendeva fino a **Chrome 36** (2015), ancora in uso su alcune versioni di TV. Questo ha comportato non solo la riscrittura completa dell'interfaccia utente, ma anche una revisione profonda della logica JavaScript, con l'introduzione di **polyfill** e l'adozione di pratiche compatibili con standard obsoleti.
- **Testing multi-dispositivo e sviluppi ad-hoc**: L'eterogeneità dei sistemi operativi e dei motori di rendering ha richiesto una fase di testing estremamente intensiva su dispositivi reali, resa possibile grazie all'accesso al laboratorio Kineton. Ogni modello ha presentato peculiarità specifiche (tempi di rendering, problemi di focus, gestione eventi) che hanno richiesto **sviluppi dedicati e correzioni ad hoc**. Questa frammentazione ha impattato significativamente sui tempi di sviluppo e testing.
- **Performance e refactoring**: Una criticità significativa è emersa dalle **performance estremamente limitate** dei dispositivi Smart TV, in particolare quelli di fascia media o datati. Questa problematica ha richiesto due cicli di refactoring completo:
- Un primo refactoring del first layer per ottimizzare la gestione del rendering
- Un successivo refactoring del second layer per garantire la scalabilità

Gli interventi hanno incluso:

- Semplificazione radicale del DOM
- Minimizzazione dei cicli di rendering
- Riduzione dell'uso di JavaScript dinamico
- Ottimizzazione di tutte le operazioni legate al ridisegno dell'interfaccia

- **Comportamenti CSS non standard**: L'approccio responsive iniziale, basato su tecniche responsive, si è rivelato inaffidabile, a causa della gestione eterogenea del layout dai motori di rendering dei browser delle Smart TV. Per seguire perfettamente il design RAI quindi è stato quindi necessario passare a una gestione **pixel-perfect**, con layout rigidi e adattati per risoluzioni fisse (1080p e 720p).

Nonostante queste complessità e l'incremento significativo dell'effort rispetto alle stime iniziali, lo sviluppo è stato portato a termine con successo. Possiamo **certificare che l'attività è conclusa**, e che eventuali rifiniture residue rientreranno nelle ore già consuntivate.

Android TV

Lo sviluppo per Android TV ha richiesto **130,5 ore**, pari a **16,3 giornate/uomo**, con una stima economica di **€8.150** (calcolata secondo la tariffa standard di €500/giornata/uomo). Il minor effort rispetto all'ambiente Smart TV è giustificato dal riutilizzo dell'interfaccia già sviluppata, che viene visualizzata tramite WebView, e del riutilizzo del pattern SDK WebView già presente su Android (mobile).

Dettaglio ore/uomo

Task: Banner su Android TV: test dell'SDK WebView, adattamenti per la compatibilità e la navigazione tramite telecomando e testing.

Risorse: Fabio Politi, Cristiano Federico, Simone Mazzotti

Fase	Fabio Politi	Cristiano Federico	Simone Mazzotti	Totale Fase
Discovery	2,0	2,5	7,5	**12,0**
Sviluppo	8,5	12,5	32,5	**53,5**
Testing e Bugfix	9,0	21,5	34,5	**65,0**
Allineamenti e Call	4,5	9,0	16,5	**30,0**
Totale Persona	**24,0**	**45,5**	**91,0**	**130,5**

Lo stato attuale vede il banner operativo e integrato in WebView all'interno di app Android TV, con interfaccia coerente, pienamente navigabile via telecomando e compatibile con le principali varianti dell’ambiente Android.

Problematiche riscontrate

Anche se l’ecosistema Android TV è meno frammentato rispetto alle Smart TV tradizionali, lo sviluppo ha comunque richiesto interventi non banali:

- **Ottimizzazione su dispositivi low-end**: Molti dispositivi Android TV (in particolare quelli con WebView non aggiornate) hanno mostrato limiti di performance significativi. Abbiamo quindi adottato tecniche di ottimizzazione specifiche, riducendo il carico sul thread UI e ridimensionando gli asset grafici.
- **Navigazione via telecomando (D-pad)**: La gestione degli input da remoto ha richiesto un **mapping personalizzato** tra eventi hardware (freccette, tasto OK, ritorno) e interazioni DOM (focus, hover, click). In alcuni casi è stato necessario sviluppare un sistema di gestione del focus completamente custom.
- **Compatibilità WebView**: Le differenze tra le versioni di WebView integrate nei vari dispositivi hanno imposto l'adozione di fallback e l'esclusione di alcune API JavaScript moderne, già sostituite nel caso Smart TV.
- **Ottimizzazione tempi di caricamento**: L'utilizzo di WebView ha introdotto un overhead significativo nei tempi di inizializzazione del banner, dovuto alla creazione e al caricamento della WebView stessa. Questo ha richiesto un'intensa attività di ottimizzazione delle performance per ridurre i tempi di apparizione del banner a livelli accettabili, intervenendo sulle performance dell'interfaccia al suo interno.

Anche in questo caso, **il risultato finale è stabile e conforme agli obiettivi iniziali**, ed è possibile considerare conclusa l'attività. Eventuali piccole correzioni o miglioramenti potranno essere inclusi nell'ambito delle normali attività di manutenzione della CMP.

HBBTV

Integrazione completata con un effort molto contenuto (**3 ore complessive**): si è trattato principalmente di uno scambio via call/mail con chi si è occupato della lavorazione

---

Attività da includere in uno step successivo

SDK Unificato: Architettura a Lungo Termine

La strategia per il futuro della CMP di Rai che proponiamo è basata sulla **combinazione React / React Native** per la realizzazione delle interfacce utente, così da consentire di "buildare" la stessa base di codice su piattaforme differenti (tvOS, Kepler, mobile, web) con il minimo effort. Di conseguenza, la logica di business viene mantenuta **headless** in **TypeScript**, fungendo da layer riusabile e indipendente e garantendo al contempo manutenibilità e scalabilità.

Obiettivo Fase Successiva: Sviluppo per tvOS e Kepler

La prossima fase di sviluppo, che richiede finanziamento, si concentrerà sulla finalizzazione della CMP per gli ambienti **Apple tvOS** e **Kepler**.

- **tvOS**: Verrà sviluppato utilizzando **React Native**, sfruttando le performance native del dispositivo.
- **Kepler (Amazon)**: Verrà sviluppato utilizzando **React Native** per garantire la piena compatibilità con il nuovo sistema operativo di Amazon per Fire TV.

Prossimi Passi

Per portare a termine lo sviluppo su tvOS e Kepler, i prossimi passi operativi sono:

1. **Discovery (Allineamento Tecnico)**: Organizzare incontri con i team di sviluppo di tvOS e Kepler per definire nel dettaglio le specifiche di integrazione del nuovo SDK e stabilire i perimetri di lavoro. Alla fine di questa fase, si dovrebbe avere una chiara definizione delle specifiche tecniche, dei perimetri di lavoro e **degli effort di sviluppo richiesti**.
2. **Sviluppo**:

- **Migrazione Logica "Headless"**: Completare il porting del core della CMP sulla nuova codebase TypeScript.
- **Sviluppo UI**: Realizzare le interfacce utente native in React Native e testarle entrambi gli ambienti.
- **Build SDK**: Creare i pacchetti nativi finali pronti per l'integrazione.

3. **Testing e Supporto**: Fornire supporto attivo ai team durante la fase di integrazione dell'SDK nelle rispettive applicazioni, garantendo che il

comportamento sia corretto e performante.

Lo sviluppo è già stato avviato con un effort di **24 ore**, che ha portato alla realizzazione di un **Proof of Concept (POC)**. Il POC ha validato la fattibilità dell'approccio React Native in ambiente tvOS, confermando la solidità della strada intrapresa.

Potenziali Sviluppi Futuri

Una volta consolidata la nuova architettura su tvOS e Kepler, si apriranno diverse opportunità per estendere la codebase unificata, con notevoli vantaggi in termini di coerenza e ottimizzazione degli effort:

- **Migrazione di Android TV**: Sostituire l'attuale implementazione basata su WebView con un layer UI nativo (React Native), riutilizzando gran parte del codice sviluppato per tvOS con un effort limitato.
- **Migrazione App Mobile (non-TV)**: Estendere l'SDK nativo anche alle applicazioni per smartphone iOS e Android, eliminando definitivamente le WebView anche in questi ambienti.
- **Integrazione completa del Web**: Unificare anche gli sviluppi per il web sotto lo stesso core logico, garantendo una coerenza totale tra tutte le piattaforme.

Questo approccio permette di capitalizzare sugli sforzi di styling e logica, rendendo la migrazione e la condivisione di componenti tra le varie piattaforme un processo a basso attrito. Le iniziative di sopra elencate non rientrano nel perimetro obbligatorio: rappresentano opzioni da valutare in base alle priorità, ma la strada scelta le permette.

Adeguamenti UI per Web e iOS

Sono inoltre previste modifiche minori all’interfaccia utente per le versioni **Web** e **iOS** della CMP, necessarie per allineare completamente la logica visuale e la user experience tra le diverse piattaforme.

Questi interventi rientreranno nel **prossimo step operativo**: in funzione dell’effort che emergerà in fase di pianificazione, valuteremo se implementarli direttamente sulla **nuova architettura React / Headless** (mantenendo la coerenza con la strategia a lungo termine) oppure mantenere l’attuale stack tecnologico. In entrambi i casi, si tratta di attività a **impatto limitato**, con l’obiettivo di garantire uniformità di stile e comportamento tra tutti i touchpoint.

Gli sviluppi portati a termine hanno richiesto un effort superiore alle stime iniziali, anche perché **Jump ha affrontato per la prima volta il mondo TV**, con una curva di apprendimento che ora rappresenta un patrimonio di competenze al servizio del progetto. Le lesson learned raccolte in questo percorso consentiranno di ridurre sensibilmente il rischio e i tempi di delivery nelle prossime fasi.

Jump conferma la volontà di voler investire sul progetto anche oltre i confini della fornitura CMP, anche affrontando attività complesse e refactor, in modo da fare evolvere il progetto a vantaggio di entrambe le parti.

Crediamo fermamente che la collaborazione fra Jump e Rai, fondata sulle competenze maturate in questa prima fase, possa portare valore aggiunto a lungo termine per entrambe le organizzazioni e per gli utenti di Rai e RaiPlay.