# STUDIO DI FATTIBILITA'

## Prototipo Cluster Digitale ICH-X K3


**Progetto:**Sviluppo Sistema Cluster Automotive

**Cliente:**ICH-X (gruppo galassia DR)

**Veicolo target:**SUV off-road "ICH-X K3"

**Motore:** Unity Engine(URP)

**Data:**Ottobre 2025

**Sviluppatore:**Gianluca Milani

---

# SOMMARIO ESECUTIVO

Andremo a realizzare un prototipo ad alte prestazioni di cluster digitale per il nuovissimo SUV ICH-X K3,orientato all’off-road.Il progetto punta ad organizzare temi,layout,varianti per modalità e cura comfort visivo in VR. Viene implementata un'architettura modulare e scalabile per permettere estensioni in futuro come ADAS o telemetria con impatti minimi

## Caratteristiche sul SUV

il **K3** è un fuoristrada di segmento D/E con lunghezza pari a 4.8m con **trazione XWD** e motore **2.0 turbo** capace di erogare **245 CV** motore,**375 Nm** di coppia,nato per uso misto strada e percorsi impegnativi

---

# REQUISITI E AMBITO

## Requisiti funzionali

- **Velocità veicolo**
  Visualizzazione tramite GaugeSpeed(Prefab:SpeedPanel.prefab)
- **RPM motore**
  Visualizzazione tramite RpmPanel.prefab con logica RpmAlertController.cs
- **Marcia inserita**
  Visualizzazione con GearLogicController.cs per le marce P/R/N/D(Prefab:ArrowGear.prefab)
- **Modalità di guida/terreno**
  Gestione tramite ScriptableObject TerrainModeSO e controller ModeButtonsController/TerrainModeController(modalità Road/Trail/Snow)
- **Tematizzazione**
  ThemeSO(Day/Night), ThemeService, ThemeApplier, ThemeSlot e controlli di aspetto (AmbientStripController).
- **G-meter**
  GMeterController.cs con GMeterPanel.prefab
- **Pressione pneumatici**
  TyrePressureController.cs e TyrePressureView.cs (Evento OnTyrePressuresChanged)
- **Inclinometri**
  - Roll:InclinometerRollController.cs

    - Pitch:InclinometerPitchController.cs (gestito/simulato lato controller; non esposto come proprietà nel servizio).

## Requisiti non funzionali

- Modularità: estendere/ sostituire pannelli e feature senza impatti sui moduli esistenti (prefab e controller isolati).
- Disaccoppiamento & testabilità: interfacce di servizio e Service Locator (registrazioni in Kernel) + EventBus per pub/sub.
- Caricamento dinamico: uso di Addressables per istanziare UiRoot all'avvio.
- Prestazioni UI: animazioni leggere su RectTransform/CanvasGroup (tween custom UIAnimator), attenzione a GC e batching sprite.
- Input: file InputSystem_Actions.inputactions presente per futura integrazione; allo stato attuale i controller principali usano KeyCode.

## Vincoli

- Engine:Unity
- Struttura asset:
    - Scena di ingresso:Assets/Scenes/entry.unity
    - Prefab UI in Assets/Resources... e Assets/Resources_moved/UI/Prefabs/.
    - Script funzionali in Assets/Scripts/MyGameFeatures/ClusterFeature/....
    - Servizi e bootstrap in Assets/Application e Assets/Domain
    - Dati Addressables in Assets/AddressableAssetsData/(chiave UiRoot utilizzata in Kernel)
- Architettura runtime: Bootstrap->Kernel.Initialize()->registrazione servizi (IEventBus, IBroadcaster, IVehicleDataService, IThemeService, …)->init Addressables->InstantiateAsync("UiRoot")->DontDestroyOnLoad->pannelli che si sottoscrivono al bus/servizi

---

# ARCHITETTURA SOFTWARE

## Pattern & Layers

**-Client–Services–Features** in Unity. Event Bus per telemetria e stati; controller di modalità per Road/Trail/Snow (state machine non centralizzata);Service Registry/Locator per disaccoppiamento
**-Data Abstraction:** VehicleDataService (Speed, RPM, Roll, G (Vector2); Pitch gestito lato widget)
**-Asset Config:** ThemeSO (Day/Night), TerrainModeSO (Road/Trail/Snow)

## Performance / Rendering

Target fluido per UI automotive. Animazioni su RectTransform/CanvasGroup tramite UIAnimator custom,batching sprite standard Unity

## Flusso di avvio

Bootstrap → attiva Kernel
Kernel.Initialize():

- registra i servizi (IEventBus, IBroadcaster, IVehicleDataService, IThemeService, ecc.)
- inizializza Addressables
- InstantiateAsync("UiRoot") e DontDestroyOnLoad.
  UiRoot contiene i vari Panel (prefab) che si sottoscrivono al bus/ai servizi

DIAGRAMMA MERMAID RUNTIME E BOOTSTRAP
DIAGRAMMA MERMAID TEMA DAY E NIGHT
DIAGRAMMA MERMAID MODALITA' DI GUIDA
DIAGRAMMA MERMAID CICLO DATI VEICOLO

---

# FEATURES SVILUPPATE E FUNZIONALITÀ

## Features Automotive Core

### Welcome/Bootstrap

- Schermata di avvio del cluster con inizializzazione dei servizi di base
- Transizione automatica alla dashboard principale al termine del bootstrap
- Caricamento pannello UI root e registrazione all'EventBus per la propagazione degli eventi applicativi

### Drive Mode (Modalità di guida)

- Set di modalità road(base),trail,snow con selezione diretta dalla UI dedicata
- Selezione rapida da tastiera tramite tasti funzione F1/F2/F3 per cambio modalità
- Aggiornamento dinamico della UI:badge e indicatori di modalità si aggiornano in tempo reale.Elementi ambientali si adeguano al profilo selezionato

### Strumentazione:Tachimetri e marce

- Tachimetro VELOCITA':visualizzazione della velocità in km/h con gauge dedicato.Integrazione con indicatori ambientali per feedback visivo
- Tachimetro RPM:gauge dedicato per il regime motore.
- Cambio marce:indicazione dello stato del cambio P/R/N/D con le conseguenti marce al cambio di velocità.
- Frecce direzionali:
  z=freccia sinistra,x=quattro frecce,c=freccia destra

### Telemetria Off-Road:G-Meter e Inclinometri

- G-Meter:pannello dedicato per accelerazioni longitudinali e trasversali,utile soprattutto in contesti off-road
- Inclinometro Pitch e Roll:pannelli separati per beccheggio e rollio, con aggiornamento continuo per il monitoraggio degli angoli del veicolo

## Features Utility

### Tire Pressure

- Monitoraggio pressione pneumatici per le 4 ruote.
  - Gonfia:2,6 bar
  - Sgonfia/bucata:1,0 bar
- Simulazione gonfiaggio/sgonfiaggio da tastiera con ruote casualmente coinvolte:b=sgonfiaggio ruota,v=rigonfiaggio ruota

### Tema Day/Night

- Toggle istantaneo Giorno/Notte con F4.
  Cosa cambia:palette UI,luminosità/contrasto

---

# IMPLEMENTAZIONE TECNICA E TESTING

## Qualità del Codice

-Logging mirato con Debug.Log sui punti critici
-Asincronia efficace: uso combinato di async/await e coroutine
-Error handling da rafforzare: molti try, pochi catch/LogError
-Gestione asset moderna con Addressables(niente Resources.Load)

## Sistema di debug basato su input da tastiera

### Modalità/Tema

- **F1/F2/F3:**Cambio modalità guida:Road,Trail,Snow
- **F4:**Toggle Day/Night

### Strumentazione–Tachimetro

- **1,2,3,4,5,6:**Limite velocità 30,50,70,90,110,130 km/h

### Controllo veicolo

- **Freccia Left/Right/UP/Down:**Controllo direzionale dell'autovettura

### Indicatori di direzione

- **Z:**Freccia sinistra
- **X:**Quattro frecce
- **C:**Freccia destra

### Simulazione pendenza / assetto

- **L,K:**Discesa e Salita
- **J,H:**Rollio(angolo)

### Cambio

- **P,N,D,R:**Selettore marce

### Pressione pneumatici

- **B,V**:Sgonfiaggio / Gonfiaggio pneumatici

---

# PIANO DI TEST

### Avvio

- Caricamento UiRoot via Addressables da entry.unity=nessun errore Console, DontDestroyOnLoad
- Servizi registrati (EventBus, VehicleDataService, ThemeService, ecc.)=disponibili ai pannelli

### Modalità (Road/Trail/Snow)

- Switch bottoni=stato e badge corretti,nessun flicker
- Switch ripetuti in movimento=stato stabile

## Tema (Day/Night)

- Toggle=tutti i ThemeApplier/ThemeSlot aggiornano colori/contrasto
- Con più pannelli visibili=AmbientStripController leggibile

## Dati veicolo

- Cambio marcia=GearLogicController,P blocca velocità,R limiti reverse
- Speed/RPM:Accel/coast/brake=SpeedPanel e RpmPanel/RpmAlertController coerenti
- G-meter:Sterzo con accel=GMeterPanel varia correttamente
- Inclinometri:Roll da service,Pitch da controller=clamp ai range
- Frecce:TurnSignals lampeggio regolare

## Pressione pneumatici

- TyrePressureController + TyrePressureView → mostra FL/FR/BL/BR (bar).
- Demo B/V=evento OnTyrePressuresChanged, UI aggiorna subito

## Limite velocità (Speed Panel)

- Tasti 1–6=imposta limite 30/50/70/90/110/130; lo speed limit mostrato si aggiorna
- Superamento=se velocità>limite,indicatore/controllo velocità diventa rosso,rientra al normale sotto limite
- Soglia=a=limite non rosso,a>limite rosso
- Persistenza=limite e stato colore restano corretti dopo Day/Night e Road/Trail/Snow

## Soglia RPM (Rpm Panel)

- Soglia=se RPM>4500, il pannello/indicatore diventa rosso; torna normale a≤4500
- Coerenza UI=colore rosso visibile su RpmPanel/RpmAlertController senza flicker
- Interazioni=comportamento invariato durante toggle tema e switch modalità

## Event Bus/Lifecycle

- Abilita/disabilita pannelli=niente handler “fantasma”
- Ordine init variabile=nessun NullReference,riallineo quando service è pronto

## Usabilità

- UX-01: Leggibilità Day/Night (contrasto, luminanza)

---

# ALLEGATI PER FONTI VEICOLO

Per raccogliere i dati tecnici come la velocità massima,il peso.. ma soprattutto per stimare il comportamento in accelerazione nei vari range di velocità utili al prototipo,ho consultato le seguenti fonti esterne:

- **Sito ufficiale K3** *(ICH-X)*
- **Pagina dealer** *(Gino Spa)*
- **Prova/recensione***(Motor1)*

---

# CONCLUSIONI E RACCOMANDAZIONI

- **Base per K3 in produzione**:pannelli modulari (Speed/RPM/Gear/G-meter/Pitch/Roll/Mode),tema Day/Night,limiti velocità e soglia RPM
- **Template riusabile ICHX**:Service Registry+EventBus,Addressables,UI a prefab/feature isolati
- **Banco prova rapido**:simulazione veicolo(PRND,limiti, alert colore, modalità terreno) per test e iterazioni veloci

## Next Step Possibili

**-Stato trazione 2H/4H/4L+differenziali**
Indicatori semplici e sempre visibili delle modalità di trazione e dei blocchi

**-Vitals motore+fuel/range/trip**
Schermata essenziale con temperature/pressioni motore,livello carburante,autonomia e contachilometri di viaggio A/B

**-Bussola+altimetro**
Orientamento immediato e quota per la guida off-road

**-Avvisi veicolo base**
Spie chiare per porte/cofano/bagagliaio,cinture e freno stazionamento,per sapere subito se si può partire