Come far funzionare in OR activity MSTS

Una piccola guida per gli sviluppatori di activity

Rel. 1.3 30/4/2014 by Carlo Santucci

Motivo e scopo del documento

Nel momento in cui sto scrivendo questo documento, l'area in cui c'è il maggior numero di incompatibilità fra MSTS e OR è quella delle activity. Activity MSTS anche di media complessità girano spesso in OR in modo significativamente differente, e perciò spesso perdono in attrattività (perchè certe “combinazioni” di treni non funzionano), o possono anche portare a stalli (cosa che non succede spesso).

Le activity girano in modo diverso su OR in parte per la difficoltà di simulare le stesse temporizzazioni di MSTS, ma principalmente per decisioni che sono state prese (almeno al momento) dagli sviluppatori di OR.

Considerando l'importanza delle attività per il divertimento nel gioco, e considerando la grande quantità di activity MSTS già sviluppate e funzionanti, è evidente che mettere a disposizione una procedura ragionevole per portare su OR activity sviluppate per MSTS può portare alla disponibilità di un buon numero di activity OR, per il piacere degli utilizzatori di OR.

Scopo di questo documento è descrivere questa procedura.

E' chiaro che la migliore soluzione è che sia lo stesso sviluppatore dell'activity MSTS a portarla su OR, e ancora meglio se egli pubblica allo stesso momento le due versioni dell'activity. Tuttavia, una persona che “conosce” l'activity originale MSTS potrebbe portarla su OR anche se non ne fosse l'autore originale.

Quanto scritto in questo documento potrebbe anche essere applicato per scrivere activity che girino solo su OR.

Attenzione: questo documento è relativo allo stato di OR alla sua release X2194. Dato che OR è un simulatore in sviluppo, parti di questo documento potrebbero non essere più applicabili per versioni future. Comunque il sottoscritto fa il possibile perchè questo non succeda.

Questo documento è stato scritto dopo avere utilizzato un'activity MSTS esistente come caso di studio: gentilmente apart mi ha posto ha disposizione una delle sue activity. L'ho portata su OR, in modo da avere le basi per questo documento. Questo primo caso di studio non comprende manovre. Un secondo caso di studio è previsto utilizzando un'activity che comprende manovre.

Mettere in piedi l'ambiente di passaggio da MSTS a OR

Come regola generale, l'attività passata ad OR sarà basata su un insieme di files parzialmente nuovi. Si suggerisce che questo nuovo insieme di files abbia il nome che comincia con OR, così che sia facile separarli dagli originali. I files originali non devono essere modificati, cioè per girare sotto MSTS vengono usati I files originali.

Portare un'activity a OR richiede un ambiente per modificare e per testare le activity.

Al momento attuale l'activity editor (AE) di MSTS è utilizzato per modificare l'activity, mentre OR è utilizzato per testarla.

Va fatto notare che l'AE di MSTS e OR possono essere aperti nello stesso momento, riducendo quindi significativamente il tempo per passare dall'uno all'altro.

OR ha una funzione piuttosto simile a quella fornita da MSTS per simulare l'esecuzione delle activity: questa funzione è la finestra DCO (attivata da CTRL-9). La “velocità” del tempo può essere aumentata (Ctrl-Alt-9(Tnum)) più e più volte, può essere ridotta (Ctrl-Alt-3(Tnum)), e può essere riportata al valore normale (Ctrl-Alt-7(Tnum)).

La funzionalità principale che manca a OR rispetto all'AE di MSTS è di avere il Player train che si muova automaticamente. Ci si augura che questa funzionalità venga implementata nel futuro (forse anche con la possibilità di commutare avanti e indietro fra treno guidato dal giocatore e treno guidato automaticamente).

Dato che non è pratico di guidare a mano il player train ad un'alta “velocità” del tempo, si suggerisce l'organizzazione seguente.

Se Train Store è disponibile, si suggerisce di utilizzarlo e di selezionare l'activity che si vuole portare su OR.

L'attività originale è aperta con l'AE (supponiamo che si chiami origact.act).

Nel “Display name” il nome dell'activity è cambiato aggiungendoci davanti OR, e ponendo alla fine la stringa “_dummyplayer”, e l'activity è salvata col nome file Ororigact_dummyplayer.act.

Si annota l'orario del player train su un file o un pezzo di carta.

Si genera una nuova versione del traffico utilizzando l'originale come modello (“template”), aggiungendo un “OR” davanti al nome e una stringa “_dummyplayer” alla sua fine.

Si crea nel traffico un nuovo servcizio usando come template il servizio usato per il player train (supponiamo che si chiami origplayertrain) e denominandolo Ororigplayer train; questo treno è aggiunto al traffico, con la stessa ora di partenza e introducendo lo stesso orario di origplayertrain.

Si crea un nuovo player train “finto”, e lo si chiama dummyplayer, e lo si sostituisce a quello originale. Si suggerisce che abbia un path breve in un binario laterale inutilizzato vicino al punto di partenza di origplayertrain. L'activity è salvata.

A questo punto abbiamo un'activity che si può testare con OR.

Si lancia l'activity con OR. Si apre la finestra DCO, e l'activity può essere fatta girare la prima volta (accelerando il tempo per fare più in fretta) e osservata, notando cosa non funziona come in MSTS.

Una cosa in più rispetto a MSTS è che si può fisicamente vedere Ororigplayertrain nella finestra principale di OR. Per fare questo, si mette un momento in pausa OR (Esc), si seleziona la camera 2 se non già selezionata, si zoomma nella finestra DCO finchè Ororigplayertrain è mostrato come un “serpente” e non come un quadrato, si clicca sulla sua loco finchè il serpente non diventa rosso, e si clicca su “Vedi in gioco”. Apparirà origplayertrain nella finestra principale (eventualmente dopo un po' di tempo se il treno è distante dal punto visualizzato in quel momento nella finestra principale di OR). Se il tempo è molto accelerato, lo scenario non sarà visualizzato del tutto, ma non è un problema.

Un'altra funzionalità utile è di premere nella finestra principale più volte Shift-F5 finchè compare la “dispatcher information” che contiene molte informazioni utili, anche se non immediatamente decodificabili.

Una spiegazione dettagliata del contenuto della “dispatcher information”, in inglese, si può trovare qui:

http://www.elvastower.com/forums/index.php?/topic/24054-dispatcher-information-explanation/page__view__findpost__p__147591

Ora occorre modificare l'activity in modo che si comporti come desiderato.

Occorre ricordare che non bisogna fare modifiche ai files originali. Ogni volta che è necessario modificare un file per OR, che sia un file .pat, .srv o anche .con, occorre farne una copia che si farà precedere dalla stringa “OR” per lasciare il file originale non modificato.

Qui di seguito è riportata una tabella che evidenzia nelle prime due colonne le differenze fra il comportamento delle activity in MSTS e in OR. Nella terza colonna si suggerisce come emulare sotto OR il comportamento in MSTS.

Se un argomento non è presente nella tabella, in generale significa che funziona nello stesso modo in MSTS e in OR (o che ho dimenticato di nominarlo...).

A questo punto si può iniziare un processo iterativo editando l'activity con l'AE di MSTS, testandolo tramite la finestra DCO, ritornando all'AE per le modifiche e così via.

Quando il comportamento dell'activity è soddisfacente, si salva una copia della stessa rimuovendo la stringa “_dummyplayer” dal nome file dell'activity e dal nome visualizzato. In questa activity si genera un nuovo traffico utilizzando quello con la stringa “_dummyplayer” e dandogli il nome senza la stringa “_dummyplayer”.

Si sostituisce il player service “finto” con Ororigplayertrain preso dal traffico e si copia l'orario da quello presente nella copia del service presente nel traffico. Infine, la copia di ORorigplayertrain presente nel traffico è cancellata.

A questo punto abbiamo un'activity che possiamo testare in OR guidando noi il player train. Può succedere che sia necessaria qualche iterazione anche per questa activity finale per tarare finemente le temporizzazioni.

Si ricordi che anche in questo caso si può “velocizzare” il tempo per fare più in fretta, anche se lo si può fare in modo limitato a causa dei tempi di reazione umani limitati...

Quando è terminata anche la taratura fine l'activity può essere resa disponibile in forma .apk o raccogliendo tutti I files il cui nome comincia con “OR” nelle directory CONSIST, ACTIVITIES, SERVICES, PATHS e TRAFFIC, con l'esclusione di quelli il cui nome termina con “_dummyplayer”.

Tabella delle differenze fra MSTS e OR

MSTS	OR	Come compatibilizzare per OR
I treni AI possono iniziare la simulazione anche con velocità >0	I treni AI iniziano la simulazione sempre a velocità =0	Anticipare l'orario di partenza del treno AI
Il player train può iniziare la simulazione anche a velocità >0	Il player train non può iniziare la simulazione a velocità >0	Cambiare le tempistiche. Attenzione: se si cambia l'ora di inizio dell'activity, tutti I tempi di tutti I treni verranno anche loro cambiati!
I treni AI non considerano gli orari impostati	I treni AI considerano gli orari impostati.	Questo è un vantaggio di OR e può essere usato spesso in luogo dei waiting point per sincronizzare i treni fra loro.
Il tempo di fermata di un player train passeggeri è il minimo fra la differenza dell'ora programmata di partenza e quella programmata di arrivo, e un valore derivato dal numero di passeggeri sul marciapiede come definito nell'AE; indipendentemente dal risultato di questo confronto, comunque, il treno non partirà prima della sua programmata di partenza. Se lì'ora programmata di partenza è = all'ora programmata di arrivo?	Se l'ora programmata di partenza è> dell'ora programmata di arrivo: Il tempo di fermata di un player train passeggeri è il minimo fra la differenza dell'ora programmata di partenza e quella programmata di arrivo, e PlatformMinWaitingTime () in secondi, come definito nella creazione della route e come può essere trovato nel file .tdb (valore di default è 180); indipendentemente dal risultato di questo confronto, comunque, il treno non partirà prima della sua programmata di partenza. Se l'ora programmata di partenza = ora programmata di arrivo: il tempo di fermata è	Si suggerisce di non definire l'orario programmato di partenza = all'ìorario programmato di arrivo, per evitare una dipendenza da PlatformMinWaitingTime ( ) .

MSTS	OR	Come compatibilizzare per OR
	PlatformMinWaitingTime (), e di nuovo comunque il treno non partirà prima dell'orario programmato di partenza.	
Il tempo di fermata di un treno passeggeri AI è un valore derivato dal numero di passeggeri sul marciapiede come definito nell'AE e non dipende né dagli orari programmati di arrivo e partenza né dall'orario di arrivo.	Come per il Player train	In generale OR si comporta meglio qui. Inserendo l'ora di partenza desiderato nell'orario del treno AI e inserendo un'orario di arrivo desiderato più basso, i treni AI partiranno all'ora di partenza desiderata (se sono arrivati prima dell'ora di arrivo desiderato).
La velocità massima di un treno AI può essere modulata col parametro “Default performance” nell'AI, anche in modo differenziato per le varie sezioni della route.	Non disponibile.	La velocità massima del treno AI in metri/secondo può essere definta editando manualmente il primo parametro di MaxVelocity () nel file .con del treno AI. Rallentamenti possono anche essere conseguiti inserendo waiting point o stop a stazioni.
Si possono usare “Passing path” per gestire gli incroci in linee a semplice binario. Tuttavia il comportamento non è sempre predicibile e il ruolo degli “optional path” non è chiaro.	I passing path funzionano in modo affidabile. Inoltre è disponibile un'ulteriore opzione sperimentale (“Usa elaborazione legata al tracciato per i path di incrocio”) che permette che i passing path definiti per il player train siano disponibili anche a tutti gli altri treni. E' considerata anche la lunghezza dei treni.	Non c'è bisogno di compatibilizzare. In questo OR funziona meglio di MSTS.

MSTS	OR	Come compatibilizzare per OR
I treni AI possono essere creati ovunque nella route (solo la creazione “sopra” un treno esistente non è permessa).	I treni AI non possono essere creati su una sezione di binario già riservata da un altro treno. In questo caso il treno AI ècreato solo quando la sezione viene liberata.	Anticipare la creazione del treno AI allo stesso punto e poi porre un waiting point o uno stop in una stazione per risincronizzare il treno, o creare il treno in un altro punto cambiando orario di partenza, se possibile.
I “reverse point” sono soggetti al bug del “reverse point che cammina”: se la distanza fra il reverse point e lo scambio che precede il reverse point è più breve che la lunghezza del treno, il treno non raggiungerà mai il reverse point, che continuerà a muoversi in avanti.	Non esiste il bug del reverse point che cammina.	Non occorre fare niente.
I “reverse point” sono considerati raggiunti quando il player train li raggiunge fisicamente.	I reverse point sono considerati raggiunti appena che il player train ha liberato del tutto lo scambio che precede il reverse point.	Non occorre fare niente.
Gli incroci su linee a semplice binario sono gestiti tramite waiting point, molte prove e evitando gli stalli che MSTS talvolta genera.	Gli incroci dei treni non soffrono di stalli grazie al meccanismo della “trappola degli stalli”, e sono gestiti al meglio definendo l'orario per i treni AI ogniqualvolta possibile. Gli stalli sono possibili solo se ci sono waiting point nella sezione di una route a singolo binario fra 2 stazioni. Gli incroci in alcuni casi non funzionano se I due treni hanno punti di partenza ad una breve distanza (in termini di sezioni) l'uno	Utilizzare orari per i treni AI se possibile, e non si avranno problemi per gli incroci. Allontanare i punti di partenza di treni incrocianti per evitare un cattivo funzionamento degli incroci, se i treni avevano punti di partenza ad una breve distanza fra loro (in termini di sezioni). Put starting points of meeting trains further away to avoid bad working of train meets when trains are created with few sections distance one of the other.

MSTS	OR	Come compatibilizzare per OR
	dall'altro.	
Le precedenze sono gestite o tramite doppio reverse point o tramite CD-service. Spesso difficile da far funzionare.	Non sono sicuro se il doppio reverse point funziona. I CD-service funzionano di solito (anche se la temporizzazione talvolta va leggermente modificata), ma potrebbero non funzionare se il loro path ha davanti a sé uno scambio preso di calcio e già posizionato correttamente per il path. The starting signal at a train station stays always closed until 2 minutes before train start.	Se ambedue i treni si fermano alla stazione, porre l'orario di partenza del treno sorpassato a un valore più alto dell'altro treno, e assicurarsi che il treno sorpassante sia già fermo alla stazione al momento in cui il segnale del treno da sorpassare può essere aperto (due minuti prima dell'orario reale di partenza). Se il treno che sorpassa non ferma alla stazione, assicurarsi che abbia già la sezione di binario dopo la stazione a lui assegnata al momento che il treno da sorpassare possa aprire il suo segnale (due minuti prima dell'orario reale di partenza). Piuttosto facile da fare.
L'aggancio passivo (il treno AI si muove e si aggancia con il player train) funziona se sono prese certe precauzioni.	L'aggancio passivo non funziona. La cosa migliore che si possa ottenere è che il treni AI si avvicini all'altro treno e si fermi lì.	Purtroppo non c'è un modo per ottenere questa funzionalità. L'aggancio passivo può solo essere sostituito da un aggancio attivo.
I waiting point in generale funzionano come programmati nell'AE (a parte i bachi).	Sfortunatamente i waiting point in generale vengono eseguiti in una posizione differente da quella di MSTS; in alcuni casi non sono eseguiti (il treno ci passa sopra senza fermarsi), ed in alcuni casi possono causare stalli.	Vedi casi particolari sotto. Qui c'è qualche mal di testa.
I waiting point nell'ultima sezione del path funzionano.	I waiting point nell'ultima sezione del path (ad esempio proprio davanti ad uno scambio preso da dietro e posizionato	Estendere il path oltre lo scambio (o oltre il segnale se c'è) se possibile, o sostituire il waiting point con una fermata a stazione se si è su un binario di fermata a stazione. Non è sempre

MSTS	OR	Come compatibilizzare per OR
	correttamente per il path) non sono eseguiti. Il treno prosegue fino alla fine del path e scompare.	possibile.
Il waiting point è eseguito dove è stato posizionato con l'AE.	Se il waiting point è eseguito, viene eseguito proprio prima del termine della sezione di binario (segnale, scambio). Se c'è più di un waiting point sulla stessa sezione di binario, vengono fusi in uno solo.	Se era usato come stop di stazione, sostituirlo con uno stop di stazione nell'orario. Altrimenti bisogna prenderlo come è.
Se un waiting point è fra due stazioni in una route a singolo binario non sorgono problemi particolari.	Ise un waiting point è fra due stazioni in una route a singolo binario, possono avvenire stalli.	Evitarli ponendo i punti di sincronizzazione all'interno delle stazioni.
Non ci sono segnali forzati dalla simulazione.	I segnali di partenza delle stazioni sono forzati sul rosso fino a due minuti prima della partenza del treno, il che in alcuni casi non corrisponde ai treni reali, e sulla GE-LI mi ha portato anche portato ad uno stallo.	Deve essere preso come è, e ha vantaggi per gestire incroci e precedenze. Ho risolto lo stallo generando una versione OR del file sigscr.dat della route (si sarebbe anche potuto risolvere con una versione OR della route, ma questo ovviamente non è consigliato).
I treni AI cominciano ad accelerare appena la loco di testa passa il cartello di velocità o il segnale che permettono una velocità maggiore.	I treni AI cominciano ad accelerare solo dopo che la coda del treno ha passato il cartello di velocità o il segnale che permettono una velocità maggiore.	OR si comporta in modo più aderente alla realtà qui. Tuttavia, per mantenereee la compatibilità nelle temporizzazioni, potrebbe essere necessario modificare l'ora di partenza o di fermata alle stazione del treno AI, o la velocità massima del treno AI come definita nel primo parametro di MaxVelocity() nel file .con del treno.
Il player train può accelerare (con il	Il player train non ha il permesso di	OR si comporta in modo più aderente al reale qui. Tuttavia, per

MSTS	OR	Come compatibilizzare per OR
limite di velocità più alto indicato nel Track monitor-F4) appena la loco di testa passa un cartello di velocità o il segnale che permettono una velocità maggiore.	accelerare finchè la fine del treno non ha passato il cartello di velocità o il segnale. Il track monitor indicherà la velocità maggiore lungo il binario al punto in cui è posizionato il cartello o il segnale, ma alzerà il limite di velocità solo quando tutto il treno avrà passato quel punto.	mantenere la compatibilità nelle tempistiche, potrebbe essere necessario modificare l'orario di partenza del player train, o il tempo di fermata nelle stazioni. ,
I treni AI possono passare i segnali permissivi chiusi, ma se un treno AI si avvicina al treno davanti a sé (sia esso un treno AI o il player train), questo treno davanti è completamente ignorato e il treno AI gli passerà attraverso. Generalmente l'activity terminerà con l'errore che il player train è deragliato.	I treni AI posso passare segnali permissivi chiusi e manterranno una distanza predeterminata dal treno davanti. Se il treno davanti si muove, il treno che segue adatterà la sua velocità in modo da seguirlo a distanza fissa.	OR si comporta meglio in questo caso. In generale una activity MSTS non dovrebbe essere costruita in modo da avere questo cattivo comportamento, quindi non c'è necessità di modifiche portando a OR l'activity.

Sigscr.dat e sigcfg.dat

Ci sono casi in cui OR richiede files sigscr.dat e/o sigcfg.dat leggermente modificati per operare correttamente.

Nella route GE-LI ho avuto un caso in cui l'uso (o il cattivo uso) di tali files causava un rosso eterno, dovuto al fatto che OR forza il rosso ai segnali

I had a case where the use (or misuse) of a couple of such files in a route caused neverending reds, due to the fact that OR forces reds in station departure signals if the train has a stop.

Talvolta nei client di OR MP (multiplayer) sono visualizzati segnali con luci spente. Questo succede se nel file sigscr.dat non sono definiti SignalAspects di default per Stop, Approach_1 e Clear_2 (perchè normalmente tali aspetti sono calcolati da sigscr.dat) e contemporaneamente se questi segnali sono forzati tramite la finestra DCO. In questo caso tali SignalAspects di default vanno aggiunti, ad esempio come sotto:

```
SignalAspects ( 3
        SignalAspect ( STOP                 "Rosso"         )
        SignalAspect ( APPROACH_1                           "Giallo")

        SignalAspect ( CLEAR_2                  "Verde"
)
)
```

Inoltre occorre ricordare che OR considera più seriamente che MSTS il parametro
SignalNumClearAhead ( ) .

Come indicazione di massima questo parametro dovrebbe essere posto almeno a 3 per tutti I segnali di tipo NORMAL.

In tutti questi casi la migliore soluzione è che OR utilizzi un proprio set di tali files.

I files specifici per OR di sigscr.dat e sigcfg.dat, se necessari, devono risiedere – mantendo il loro nome originale - in una sottodirectory “OpenRails” nella directory radice della route. Anche se uno solo dei due files è stato modificato, ambedue devono essere presenti in tale sottodirectory (più in generale se più di un file di script è richiamato in sigcfg.dat – succede raramente – tutti tali script devono essere presenti in tale sottodirectory).

Utilizzo della finestra DCO durante la simulazione per influenzare l'activity

Si ricorda che la finestra DCO è un mezzo potente per influenzare l'esecuzione delle activity durante la simulazione, mezzo che non è disponibile con MSTS. In particolare:

- I segnali possono essere aperti o chiusi sia per il player train sia per I treni AI
- gli scambi possono essere girati sul path del player train.

Gli scambi non possono essere girati sul path dei treni AI. Se un treno AI arriva di punta a uno scambio che è posizionato nella direzione divergente rispetto al suo path, il treno sparisce.

L' utilizzo “voluto” della finestra DCO, previsto dallo sviluppatore dell'activity, durante la simulazione può aumentare l'attrattiva delle activity.

Credits

- Rob Roeterdink le sue spiegazioni su come OR gestisce le activity
- apart per lasciarmi pasticciare con le sue activityfor letting me fiddling with his activities.