HAPPY CODE INC

BR-jsys *business rules* per sistemi gestionali in architettura J2EE

Analisi Dei Requisiti

Versione 2.6- 16 febbraio 2008

Capitolato: "BR-jsys"

Data creazione:	18/11/2007
Versione:	2.6
Stato del documento:	formale, esterno
Revisione RR	
Redazione:	Michele Bortolato, Alessia Trivellato
Revisione:	Marco Tessarotto
Approvazione:	Elena Trivellato
Revisione RPD	
Redazione:	Marco Tessarotto
Revisione:	Alessia Trivellato, Michele Bortolato
Approvazione:	Elena Trivellato

Lista di distribuzione

HappyCode inc	Gruppo di lavoro
Tullio Vardanega, Renato Conte	Committenti
Zucchetti S.r.l	Azienda proponente

Diario delle modifiche

Versione	Data rilascio	Descrizione
2.6	14/02/2008	Correzione sintattica e grammaticale
2.5	11/02/2008	Deprecato requisito F9 e modificato i requisiti F4 e F8
2.4	05/02/2008	Aggiunta del nome del file nel modello di documento
2.3	04/02/2008	Correzione grammaticale
2.2	22/01/2008	Modifica al layout dei documenti e prima correzione grammaticale
2.1	21/01/2008	Modifiche in conseguenza del secondo incontro con l'azienda
2.0	10/01/2008	Inserito tracciamento dei requisiti
1.4	09/01/2008	Modifiche alla struttura del documento e all'ordine dei paragrafi
1.3	03/01/2008	Modifiche ai casi d'uso e alla loro analisi
1.2	21/12/2007	Documento sottoposto a revisionamento automatico
1.1	03/12/2007	Correzioni grammaticali, correzioni strutturali
1.0	28/11/2007	Inserimento casi d'uso
0.2	22/11/2007	Aggiornamento requisiti
0.1	18/11/2007	Stesura preliminare del documento

Indice

Capitolo 1

Introduzione

1.1 Scopo del documento

Il presente documento viene redatto al fine di identificare i requisiti presenti nel capitolato d'appalto denominato "BR-jsys", conseguire il raggiungimento dell'obiettivo richiesto e soddisfare le esigenze di tutti gli stakeholders coinvolti nella realizzazione del progetto di cui sopra.

1.2 Scopo del prodotto

Il prodotto andrà inserito in un progetto più ampio. Il suo obiettivo è quello di testare la consistenza di business objects, altrimenti difficilmente verificabile tramite vincoli relazionali. Verrà creato quindi un linguaggio per la definizione di business rules. Quest'ultime verranno immagazzinate in un *file repository*; nel momento in cui l'utente avrà bisogno di verificare la consistenza di un business object, un interprete si preoccuperà di invocare ed eseguire le regole che lo riguardano, stabilendo così se esso le rispetta tutte o meno. Il progetto dovrà integrarsi infine con le componenti già sviluppate dall'azienda proponente. In particolare dovrà interfacciarsi con il loro interprete, il quale dovrà poi applicare le regole ai business objects.

1.3 Definizioni, acronimi, abbreviazioni

Nella tabella di seguito vengono riportate tutte le abbreviazioni utlizzate nel documento, comprese le sigle utilizzate per il tracciamento dei requisiti.

Abbreviazione	Significato
C04	Indica il quarto capitolato d'appalto denominato BR-jsys.
I01	Indica il primo incontro avuto con il proponente descritto nel documento Incontro2007-11-22.pdf
I02	Indica il secondo incontro avuto con il proponente descritto nel documento Incontro2008-01-17.pdf
F#	Con F seguito da un numero intero sono indicati i requisiti funzionali.
NU#	Con NU seguito da un numero intero sono indicati i requisiti di usabilità.
NPo#	Con NPo seguito da un numero intero sono indicati i requisiti di portabilità.
NPr#	Con Npr seguito da un numero intero sono indicati i requisiti di prestazionali.
NQ#	Con NQ seguito da un numero intero sono indicati i requisiti di qualità.

1.3.1 Glossario

Per gli altri termini non citati nella tabella soprastante si fa riferimento al file esterno Glossario.1.8.pdf .

1.4 Riferimenti

- Capitolato d'appalto concorso per sistema "BR-jsys";
- Verbale dell'incontro con il proponente "Incontro2007-11-22.pdf";
- Verbale dell'incontro con il proponente "Incontro2008-02-05.pdf";
- "Ingegneria del Software" 8a edizione - Ian Sommerville.

1.5 Descrizione del resto del documento

Di seguito viene descritto il sistema designato dalla sigla "BR-jsys". Viene inoltre brevemente descritta la sua funzionalità, la tipologia di utenti ed il contesto d'uso. L'analisi del sistema viene presentata con l'ausilio di un formalismo grafico: i diagrammi use-case UML. Vengono infine riportati tutti i requisiti individuati dalla *Happycode Inc.* .

Capitolo 2

Descrizione generale

2.1 Prospettiva del prodotto

Il sistema va a collocarsi all'interno di un progetto globale di automazione nella generazione del codice per la produzione di software gestionale. In particolare si occuperà della gestione delle business rules, definendo un linguaggio con cui descriverle, oltre ad un sistema di archiviazione-recupero delle stesse.

2.2 Funzioni del prodotto

Il sistema si occupa di definire un linguaggio con cui descrivere business rules. Più concretamente dovrà verificare che le business rules inserite dall'utente rispettino il linguaggio sopra citato. In caso di esito positivo dovranno essere inserite in un repository, dal quale potranno poi essere recuperate nel momento in cui si desideri verificare la consistenza di un business object.

2.3 Caratteristiche utente

Il sistema andrà inserito all'interno di un progetto più ampio, il cui utente finale è un individuo esperto in ambito commerciale e gestionale, ma non necessariamente in ambito informatico.

2.4 Vincoli generali

Il linguaggio di programmazione adottato per lo sviluppo del prodotto sarà Java. Il sistema dovrà inoltre potersi interfacciare con le componenti già sviluppate dall'azienda proponente.

2.5 Presupposti e dipendenze

- Le business rules su cui si opera saranno di tipo *Rejector*. Una business rule deve essere composta da uno o più confronti tra espressioni dello stesso tipo (eventualmente complesse).
- I business objects che manipoleremo sono generalmente costituiti da campi dati semplici (numeri reali, booleani, stringhe), da un insieme di array o vettori omogenei tutti della stessa lunghezza (derivanti da tabelle) e da riferimenti ad altri business objects (ma non da array di business objects).

NQ4 OPZIONALE :

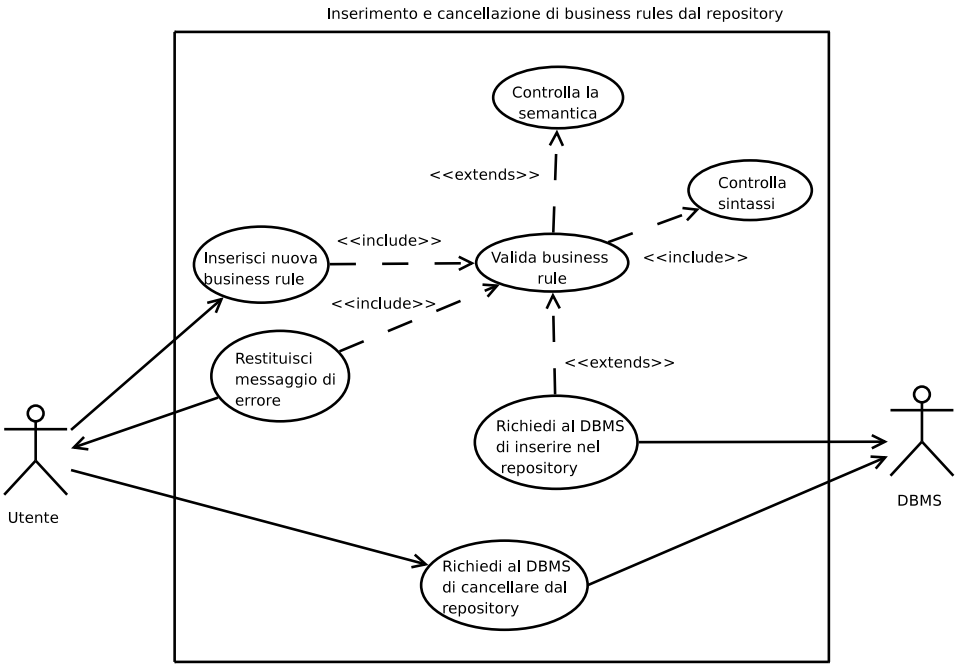
- La virtual machine utilizzata per sviluppare gli oggetti richiesti è Java 1.5.

Capitolo 3

Requisiti Specifici

Per illustrare e rendere più chiaro il sistema e le sue componenti, verranno qui presi in considerazione dei diagrammi use-case UML. Verranno elencati in seguito i requisiti individuati, ognuno dei quali preceduto da una sigla identificativa e seguito da una sigla che indica il documento da cui proviene.

3.1 Inserimento e cancellazione di business rule.



3.1.1 Inserisci nuova business rule.

Attori coinvolti: L'utente finale che desidera inserire nel repository una nuova business rule.

Scopo e descrizione sintetica: Il sistema riceve in ingresso una nuova business rule che dovrà essere validata.
Flusso di eventi: Il sistema acquisisce la nuova business rule appena inserita dall'utente e la invia successivamente al validatore.
Precondizioni: L'utente ha inserito una nuova business rule.
Postcondizioni La business rule viene inviata al validatore.

3.1.2 Valida business rule.

Scopo e descrizione sintetica: Il validatore riceve in ingresso la business rule e si preoccupa di eseguire una serie di controlli per la sua validazione.
Flusso di eventi: Il validatore una volta acquisita la business rule la sottopone a diversi controlli per poterla validare.
Precondizioni: Il sistema ha acquisito una nuova business rule.
Postcondizioni La business rule viene inviata alla componente che valida la sintassi.

3.1.3 Controlla sintassi.

Scopo e descrizione sintetica: Si controlla che la nuova business rule sia formulata secondo le specifiche e la sintassi del linguaggio scelto.
Flusso di eventi: Viene controllata la sintassi della nuova business rule appena inserita.
Precondizioni: Il validatore ha inviato la business rule al controllo della sintassi.
Postcondizioni La business rule può avanzare al successivo controllo.

3.1.4 Controlla la semantica.

Scopo e descrizione sintetica: Si controllano i tipi nella nuova business rule e si verifica che i campi dati utilizzati siano realmente presenti nel business object associato.
Flusso di eventi: Vengono controllati i tipi nella business rule e la corrispondenza tra i campi dati usati nella business rule e quelli presenti nel business object ad essa associato.
Precondizioni: Il validatore ha inviato la business rule al controllo semantico.
Postcondizioni La business rule è pronta per essere inserita nel *repository*.

3.1.5 Restituisci messaggio di errore.

Attori coinvolti: L'utente finale che ha cercato di inserire una nuova business rule e non ci è riuscito.
Scopo e descrizione sintetica: Il sistema comunica all'utente che la business rule inserita presenta degli errori.

Flusso di eventi: Il sistema a fronte di un errore nella validazione della business rule invia una segnalazione dell'errore all'utente.
Precondizioni: La validazione non è andata a buon fine.
Postcondizioni L'utente riceve un messaggio che lo informa dell'errore commesso nella creazione della nuova business rule.

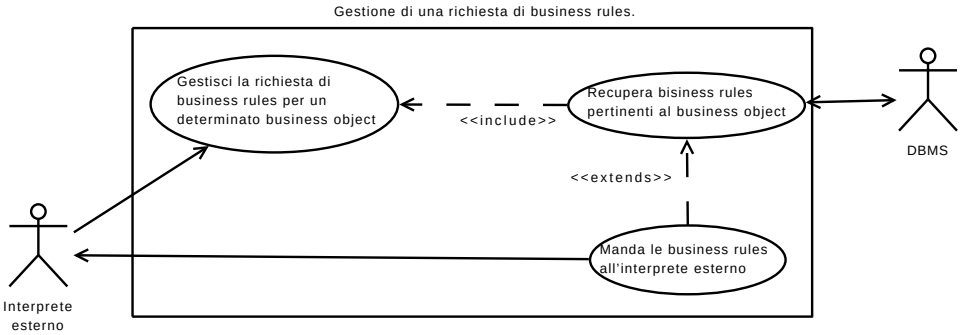
3.1.6 Richiedi al DBMS di inserire nel repository.

Attori coinvolti: Il DBMS che si preoccuperà di inserire la regola nel repository.
Scopo e descrizione sintetica: Il sistema comunica al DBMS che la business rule è stata validata e può essere inserita nel repository.
Flusso di eventi: Il sistema terminata la validazione invia la business rule al DBMS che si preoccuperà di inserirla nel *repository*.
Precondizioni: La validazione è terminata senza errori.
Postcondizioni Il DBMS inserirà la business rule nel *repository*.

3.1.7 Richiedi al DBMS di cancellare dal repository.

Attori coinvolti: L'utente finale che richiede al sistema di cancellare una business rule dal repository e il DBMS che si preoccuperà di cancellarla.
Scopo e descrizione sintetica: Il sistema mette in comunicazione l'utente finale col DBMS richiedendo al DBMS la cancellazione della business rule indicata dall'utente
Flusso di eventi: Il sistema riceve la richiesta di cancellazione di una business rule e la comunica al DBMS.
Precondizioni: L'utente ha chiesto che una determinata business rule venga cancellata.
Postcondizioni Il DBMS cancellerà la business rule dal *repository*.

3.2 Recupera business rules.



3.2.1 Gestione della richiesta.

Attori coinvolti: L'interprete esterno fornito dall'azienda che richiede la validazione di un business object.
Scopo e descrizione sintetica: Il sistema riceve una richiesta di validazione per un business object.
Flusso di eventi: Il sistema ha ricevuto la richiesta di validazione per un determinato business object.
Precondizioni: L'utente ha richiesto di validare un business object.
Postcondizioni Recupero delle regole pertinenti dal DBMS .

3.2.2 Recupera business rules pertinenti.

Attori coinvolti: Il DBMS che si occuperà di recuperare le regole
Scopo e descrizione sintetica: Il sistema chiede al DBMS di recuperare le regole dal repository.
Flusso di eventi: Il sistema richiede al DBMS di estrarre dal repository tutte le regole riguardanti quel business object.
Precondizioni: Il sistema ha acquisito la richiesta di validazione per un determinato business object.
Postcondizioni Il DBMS riceve la richiesta di recuperare delle business rules dal repository.

3.2.3 Manda le business rules all'interprete.

Attori coinvolti: L'interprete esterno fornito dall'azienda che riceve le business rules per poter validare il suo business object.
Scopo e descrizione sintetica: Il sistema ricevute le regole dal DBMS le invia all'interprete esterno.
Flusso di eventi: Il sistema ricevute dal DBMS tutte le regole riguardanti un determinato business object le invia all'interprete esterno.
Precondizioni: Il DBMS ha terminato il recupero di tutte le business rules che riguardano il business object dato.
Postcondizioni L'interprete utilizzerà le business rules appena ricevute.

3.3 Requisiti funzionali

F1 Si dovrà creare un linguaggio per la rappresentazione di business rules. [C04]

F2 Il sistema dovrà fornire un validatore che consenta l'accettazione di nuove business rules soltanto se esse sono scritte secondo le specifiche del linguaggio creato. [C04]

F3 Il sistema dovà inserire nel repository la nuova business rule dopo la sua accettazione da parte del validatore. [C04]

F4 Il sistema deve informare l'utente dell'esito della validazione di una nuova business rule. Se la validazione è andata a buon fine l'utente deve essere informato in maniera chiara del buon esito e del tempo di risposta della validazione. Se la regola non viene accettata l'utente deve essere informato in maniera chiara di dove la sua regola non rispetta le specifiche del linguaggio. [C04]

F5 Nell'interagire col repository il sistema dovrá appoggiarsi ad un DBMS esterno. [I02]

F6 Il sistema deve essere in grado di interfacciarsi con l'interprete esterno fornito dall'azienda proponente provvedendo ad esso le regole business da eseguire. [I02]

F7 Il sistema dovrà consentire la cancellazione di business rules all'interno del repository. [I02]

F8 Il sistema dovrà essere dotato di un'interfaccia grafica minimale che consenta di definire una nuova business rule e una volta validata inserirla nel repository. Tale interfaccia dovrà inoltre consentire di cancellare business rules dal repository. Tutto ciò servirà a testare in modo rapido e chiaro il corretto funzionamento del prodotto. [I02]

F9 DEPRECATO : Il validatore deve essere dotato di un metodo per il controllo di coerenza tra business rules. Esso verificherà cioè se la business rule da inserire nel repository entra in conflitto con le altre già presenti. Più dettagliatamente tale validatore dovrà verificare che le business rules presenti nel repository rappresentino un insieme soddisfacibile e non rigettino un qualunque insieme di valori possibili per i business objects. [C04]

F10 Il validatore nel validare una business rule deve poter accedere ai campi del business object ad essa associato, qualora la business rule utilizzi tali campi. [I02]

F11 Ad ogni business rule è associato un business object nel quale vanno eventualmente ricercati i campi dati utilizzati nella business rule. [I02]

3.4 Requisiti di usabilità

NU1 Il linguaggio dovrà essere di alto livello. Dovrà cioè essere semplice da capire per un utente con bassa conoscenza informatica. [I01]

NU2 Il linguaggio deve fornire funzioni primitive per le operazioni base tra i dati, come la somma, la media aritmetica e la negazione logica. [I01]

NU3 Ad ognuno dei confronti presenti in una business rule deve essere possibile, da parte dell'utente, associare un messaggio che aiuti l'utente finale (ossia quello che chiede la validazione di un business object) a capire dove la validazione ha riscontrato problemi. [I02]

3.5 Requisiti di portabilità

NPo1 Il formato di salvataggio delle business rules e del repository deve essere XML. [C04]

3.6 Requisiti prestazionali

NPr1 Il DBMS che andrà a interagire con il repository dovrà garantire una buona velocità nell'interrogazione (mediante query) del file XML, anche in presenza di un consistente popolamento dello stesso. [I02]

NPr2 Ad ogni business rule deve venire associata una stringa identificativa univoca per consentire un'efficiente indicizzazione delle business rules nel repository. [I02]

3.7 Requisiti di qualità

NQ1 Il linguaggio dovrà permettere tutte le operazioni consentite tra scalari e matrici. Il validatore di conseguenza validerà solo le operazioni consentite. [I01]

NQ2 Il progetto verrà accompagnato da un manuale che descriverà il linguaggio di definizione delle business rules. [C04]

NQ3 Il progetto verrà accompagnato dalla descrizione delle API relative al validatore e al DBMS. [C04]