UNIVERSITÀ DEGLI STUDI DI
MODENA E R EGGIO EMILIA

Dipartimento di Scienze Fisiche, Informatiche e
Matematiche

CORSO DI LAUREA IN INFORMATICA

Progetto e Sviluppo di una SAP Business Server Page Application di Supporto alla Gestione del Magazzino

Emmanuel Bonsu Boachie

Tesi di Laurea

Relatore:
Ing. Riccardo Martoglia

Anno Accademico 20017-2018

Ringraziamenti

Ringrazio l'Ing. Riccardo Martoglia per la disponibilità e la pazienza dimostrata durante questi mesi di lavoro.

Un ringraziamento particolare agli amici che mi hanno sostenuto ed aiutato costantemente.

Ringrazio Marian, Mario e Diego per il sostegno che mi hanno dato durante il periodo di tirocinio.

Infine ringrazio i miei genitori che hanno creduto in me ed hanno sempre incoraggiato a non desistere.

Parole chiave

ABAP
ABAPI
Javascript
CSS
Html
SAP

Indice

Introduzione 10
Capitolo 1 10
1.1 Presentazione 10
1.2 Obiettivi dello stage 10
1.3 Scenario esistente 11
1.4 Scenario della situazione desiderata 11
1.4.2 Soluzione informatica 11
1.5 Presentazione del progetto 11
1.5.1 Cos'è un'applicazione BSP 12
1.5.1 Cos'è un'applicazione BSP?. 12
1.5.2 Struttura di un'applicazione BSP 13
1.5.2.1 Componenti di un BSP 14
1.5.2.1 Un'applicazione BSP è costituita dai seguenti componenti ... 14
1.5.3 Accesso a un'applicazione BSP 16

Capitolo 2 19
2.1 Elaborazione di un'applicazione BSP 19
2.2 Control Flow di un BSP. 20
2.2 Control Flow di un BSP 20
2.2.1 BSP Stateless 21
2.2.2 BSP Stateful 24
2.3 Creazione di un'applicazione BSP 25
2.3.1 Pagina con logica di flusso 25
2.3.2 VIEW 26
2.3.3 Frammento di pagina 26
2.4 Procedura 26
2.5 Creare un controller 27
2.5.1 Procedura 27
2.6 Specifi dei parametri URL di sitema 27
2.7 Classi di applicazione di un BSP Application 30
2.7 Classi di applicazione di un BSP Application 30
2.7.1 Runtime Behavior 30
2.7.1 Runtime Behaviour 30
2.7.2 Accesso all'oggetto dell'applicazione 31
2.7.3 Requisiti per una classe di applicazione 31

Capitolo 3 35
3.1 Specifiche del Software 35
3.2 Requisiti Funzionali e Non Funzionali 36
3.3 Progettazione con UML 40
3.3.1 Attori del sistema 40
3.3.2 Casi d'Uso 40

CAPITOLO 4 42
4.1. MIME 43
4.1.1 Cartella CSS 43
4.1.2 Javascript 47
4.1.3 Immagine 49
4.2 Controller 50
4.3 Pagine di controllo 50
4.3.1 Altributi 51
4.3.2 Layout 52
4.3.3 Event handler 58
4.3.3.1 OnCreate 58
4.3.3.2 OnRequest 58
4.3.3.3 OnManupulation 61
4.3.3.4 OnInitialization 61
4.3.3.5 OnDestroy 62
4.3.3.6 OnInputProcessing 62
CAPITOLO 5 66
5.1 Raggiungemento degli obiettivi 66
5.1 Raggiungimento degli obiettivi 66
5.2 Conoscenze acquisite 66

CONCLUSIONE 66
Bibliografia e Sitografia 67
Acronimi e Definizioni 68

Introduzione

Il presente documento illustra il lavoro svolto durante il periodo di stage dal laureando Emmanuel Bonsu Boachie presso l'azienda REWM S.r.l.

Questa tesi ha l'obiettivo di illustrare come si estende l'interfaccia di SAP attraverso la creazione di un'applicazione SAP *Business Server Page* (BSP application) per supportare il lavoro che si svolge nel magazzino di un azienda.

SAP è un software ERP (*Enterprise Resource Planning*), Il sistema ERP dell'azienda consente ai suoi clienti di gestire i loro processi aziendali, inclusi contabilità, vendite, produzione, risorse umane e finanza, in un ambiente integrato. L'integrazione garantisce che le informazioni fluiscano da un componente SAP all'altro senza la necessità di inserire dati ridondanti e contribuisce a rafforzare i controlli di processo, finanziari, e legali. Inoltre facilita l'uso efficace delle risorse, tra cui manodopera, macchine e capacità produttive.

Il gestionale SAP offre un modulo WM " *Warehouse Management*" per un supporto flessibile e automatizzato nell'elaborazione di tutti i movimenti delle merci e nella gestione delle scorte nel proprio complesso di magazzini. Il sistema supporta l'elaborazione pianificata ed efficiente di tutti i processi logistici all'interno del tuo magazzino.

BSP l'acronimo di *Business Sever Page*, è un'estensione di SAP R / 3. Visualizza le applicazioni SAP su un browser Web o dispositivo mobile (*WAP phone*). È molto simile alla tecnologia della pagina del server e utilizza i moduli di funzione, BAPI, HTML, Javascript e ABAP per recuperare i dati dal sistema SAP. Non utilizza ITS (*Internet Transaction Server*) ma usa SAP Web AS (Application server versione 6.10).

Obiettivi

L'obiettivo dello stage era di apprendere il linguaggio ABAP e applicare ciò che è stato appreso nello sviluppare di un'applicazione *Business Server Page* (*BSP Application*), che assiste gli dipendente di un azienda a

svolgere il loro lavoro quotidiano, interfacciandosi con la gestionale(SAP) aziendale.

Struttura del documento

Il presente documento è suddivisa in due parti ed è strutturata in cinque capitoli. Nella prima parte sono spiegati come si implementa un'applicazione BSP; mentre la seconda parte è relativa all'implementazione dell'applicazione.

Il documenti e così strutturato:

Parte I: Presentazione e Introduzione al BSP

Capitolo 1: Presentazione e introduzione a BSP. Il seguente capitolo ha lo scopo di presentare, illustrare BSP, la struttura e il processo di creazione di un applicazione *BSP*.

Capitolo 2: Elaborazione e creazione di un BSP. In questo capitolo si presentano come si avviene l'elaborazione e la creazione di un applicazione BSP. Vieni inoltre spiegato come e formato l'*url* per accedere al BSP *runtime enviorment* e il concetto di incapsulamento la *Business Logic* del BSP.

Parte II L'applicazione.

Capitolo 3 –Requisiti funzionale, casi di uso: In questo capitolo si elencano i requisiti e requisiti funzionale, assieme al caso di uso dell'applicazione BSP richiesto.

Capitolo 4 - Nel quarto capitolo si illustra codice di alcuni parte del codice scritta per la progettazione dell'applicazione BSP presentato nel capitolo precedente.

C**apitolo 5** Conclusione - conterrà le conclusioni relative all'esperienza di stage svolta.

Riguardo la stesura del testo, relativamente al documento sono state adottate le seguenti convenzioni tipografiche:

• gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati vengono definiti nel glossario, situato alla fine del presente documento;

Capitolo 1

1.1 Presentazione

REWM s.r.l. è una di consulenza e formazione, nata con lo scopo di prestare consulenza alle azienda che vogliono adottare il gestionale SAP in particolare il modulo *Warehouse Management* (da cui il WM) e formare consulente in tale ambito.

L'azienda in questione offre diverse servizi ed ha contatti con varie aziende del territorio, occupandosi di consulenza informatica.

I servizi offerti da REWM s.r.l. Possono essere classificati in:

- Consulenza;
- Formazione.
- Assistenza (per la gestione del gestionale SAP all'interno delle aziende)

1.2 Obiettivi dello stage

Lo stage prevede l'inserimento dello studente nell'azienda come supporto tecnico di SAP a fianco ad un senior, apprendere il linguaggio ABAP e applicarlo nello sviluppo di un applicazione BSP SAP utilizzando diverse tecnologie che verranno descritte in questo documento.

L'applicazione BSP realizzata avrà il compito di facilitare e supportare il lavoro svolto dal dipendenti aziendale nella gestione del magazzino.

1.3 Scenario esistente

I lavoratori che gestiscono il magazzino di norma, interagiscono con il gestionale attraverso una postazione fissa ed una volta elaborate transazione, il sistema produce una stampa mediante il quale, svolgono le operazioni stampati. Dopo aver terminata le operazione, si deve sempre ritornare per rettificare eventuali operazione non fattibile a causa di errore,

quindi riconfermare il lavoro che si e potuto o meno svolgere. Quindi non esiste un interfacciamento diretto con il sistema dell'azienda, per ciò si devono sempre recarsi presso uno dei tanti postazione fissi collegati al gestionale situato nel magazzino.

1.4 Scenario della situazione desiderata

La situazione futura desiderata per la trascrizione dei documenti prevede l'interazione da parte del personale con il gestionale aziendale attraverso un palmare, un dispositivo mobile, con la capacita di leggere codice a barra in modo da automatizzare l'acquisizione di documenti o codice materiali quindi eliminare il processo di dover digitare informazione riguardante un documento (evitando cosi anche possibili errore di digitazione).

1.4.2 Soluzione informatica

La soluzione informatica proposta consiste nella realizzazione di un'applicazione Business Server Page (BSP) che estende le funzionalità del gestionale sul dispositivo mobile.

1.5 Presentazione del progetto

In questa sezione sono presentati i vari linguaggi e tecnologie impiegati nello sviluppo del BSP.

Un'applicazione Business Server Page (BSP) è un progetto di sviluppo indipendente creato e modificato nell'ambiente di sviluppo SAP (transazione SE80 come mostrato nella figura 1). È possibile utilizzare strumenti di progettazione esterni, come Adobe GoLive, Dreamweaver o Microsoft FrontPage 2000 per progettare l'applicazione BSP e le estensioni BSP.

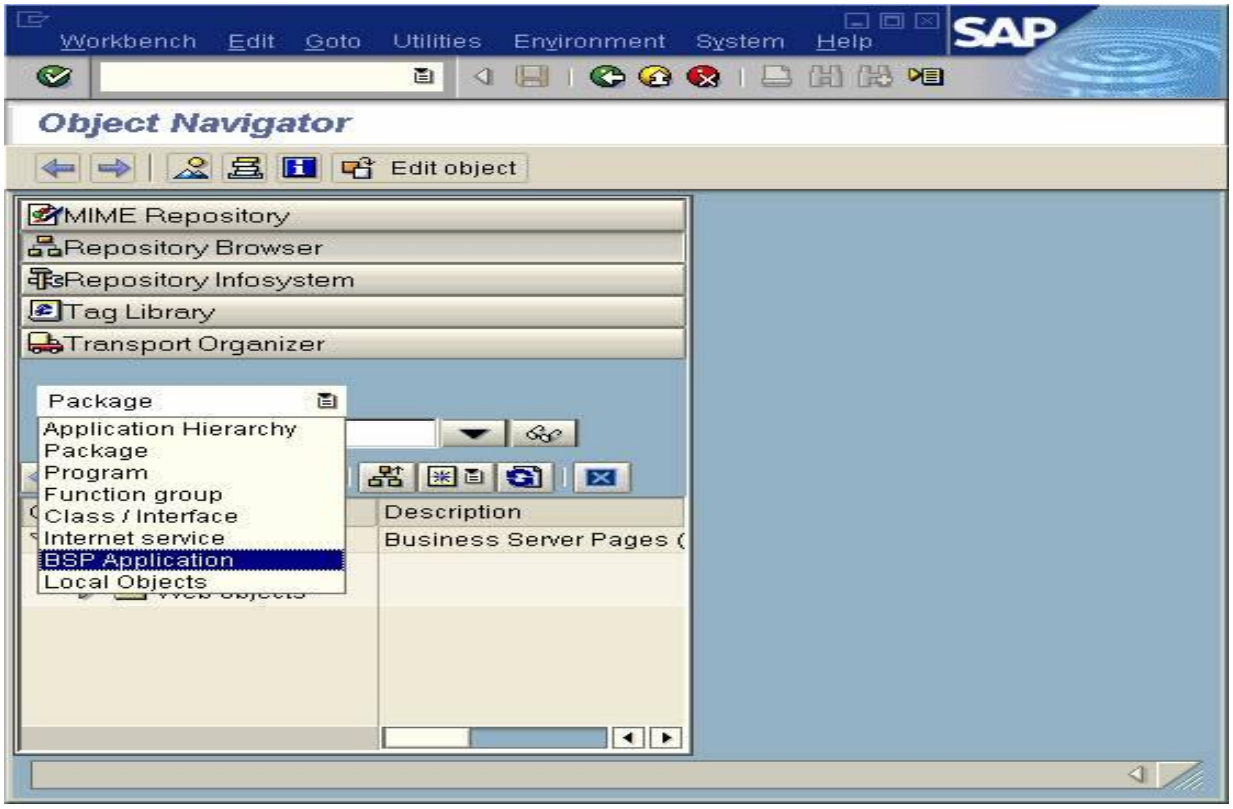

Figura 1: Menu transazione SE80, creazione applicazione BSP.

1.5.1 Cos'è un'applicazione BSP

Un'applicazione *Business Server Page* (BSP) è un'applicazione funzionale completa, come una classica transazione SAP. Tuttavia, questa applicazione viene eseguita in un browser Web anziché in SAPGUI. HTTP o HTTPS è il protocollo utilizzato per accedere all'applicazione sulla rete, consentendo l'uso di prodotti standard come firewall e server proxy.

Il modello di programmazione di *Business Server Pages* (BSP) è simile alla tecnologia della pagina del server. L'obiettivo del modello di programmazione BSP sono i punti che garantiscono una struttura ottimale nelle interfacce e nella logica di business.

1.5.2 Struttura di un'applicazione BSP

Analogico a una transazione classica, un'applicazione BSP è costituita da un'interfaccia utente e una logica aziendale assegnata ad un'unità logica.

L'interfaccia utente di un'applicazione BSP include:

- Siti Web statici.

- I siti Web generati dinamicamente, che sono BSP o modelli che contengono script sul lato server eseguiti dinamicamente in fase di *runtime* per generare un BSP del sito Web statico, possono essere pagine con logica di flusso o viste.

- Ci possono anche essere controller se viene utilizzato il modello di progettazione MVC

- Vari oggetti MIME, come immagini, icone, file audio, fogli di stile e così via, che sono parti di una tipica applicazione Web

Tutti questi oggetti sono integrati in *Change and Transport Organizer* come parti dell'applicazione BSP e sono gestiti come un'unità logica. Ciò consente a tutti gli oggetti che fanno parte di un'applicazione BSP di essere trasportati in modo completo e conforme tra tutti i sistemi in un panorama di sistema SAP.

È possibile accedere alla logica aziendale dall'applicazione BSP utilizzando i metodi tipici, come BAPI, moduli funzione o librerie di classi. Inoltre, il modello di programmazione BSP fornisce uno strumento di strutturazione, come illustrato nella figura 1.2, la classe *BSP-Application*, che può essere utilizzato per incapsulare la funzionalità della *business logic* nell'applicazione BSP.

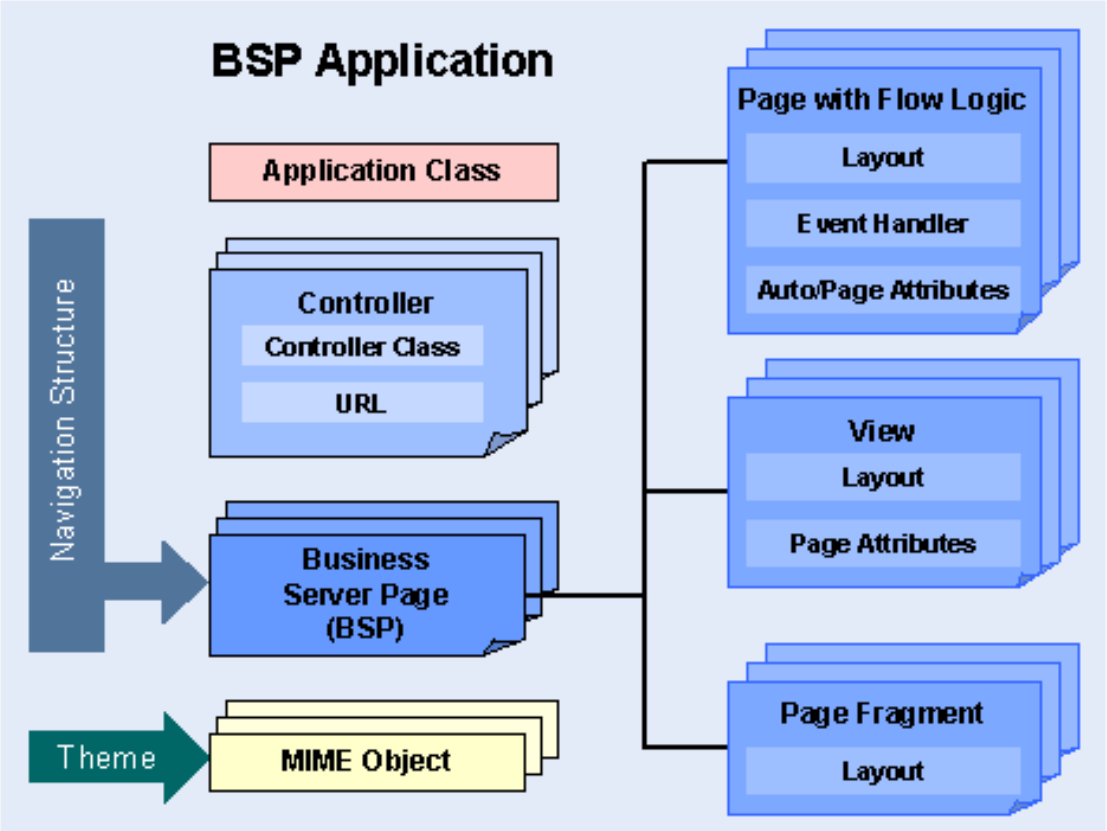

Figura 1.2: Struttura di una applicazione BSP.

1.5.2.1 Componenti di un BSP

I seguenti componenti formano un'applicazione BSP:

- **Controller**

  I controller è costituito dalla logica aziendale e i dati dell'applicazione. I controllori valutano i dati di una richiesta in arrivo in base a un modello e quindi selezionano una vista adatta per il *rendering* della risposta all'utente, come succede nella paradigma *Model View Controller* (MVC).

- **Pagine Business Server (BSP)**

  I BSP sono i siti Web visualizzati nel browser quando l'applicazione è in uso. I BSP possono contenere codice HTML statico e codice di script dinamico (ABAP o JavaScript). Il codice di *scripting* è interpretato sul server.

- **Pagina con logica di flusso**

  Si tratta di pagine semplici con gestori di eventi, ma senza molta logica di applicazione o elementi di visualizzazione. È possibile creare un'applicazione BSP esclusivamente fuori dalle pagine con logica di flusso e gestori di eventi.

- **View:**

  Come nel Model View Controller, le view sono utilizzate per visualizzare i dati.

- **Frammento di pagina**

  Questi vengono creati allo stesso modo dei normali BSP, ma vengono quindi contrassegnati come frammenti di pagina. Altri BSP possono anche includere questi frammenti usando la direttiva inclusiva.

- **Strutture di navigazione**

  La struttura di navigazione determina quale richiesta di navigazione viene utilizzata per indirizzare il processo di navigazione da quale pagina a quale pagina successiva.

- **Classe di applicazione**

  La logica aziendale di un'applicazione BSP è racchiusa in una classe di applicazione. Questa classe viene realizzata per mezzo di una classe ABAP globale che implementa l'accesso ai dati aziendali utilizzando le chiamate BAPI, ad esempio. Ogni pagina di un'applicazione BSP può fare direttamente riferimento ai componenti di questa classe (attributi, metodi e così via) utilizzando l'applicazione Object predefinita.

È anche possibile assegnare diverse applicazioni BSP a una classe di applicazione.

❖ **Oggetti MIME**

Nel sistema SAP, tutti i MIME, come la grafica, i fogli di stile (usati per definire le proprietà di formattazione dei singoli tag HTML), i file audio, i file video e così via, vengono archiviati e amministrati in un *repository* centrale, il *repository MIME*.

Per ogni nuova applicazione BSP, viene creata una directory con lo stesso nome nel *repository MIME*. Questa directory viene utilizzata come posizione di archiviazione per tutti i MIME specifici dell'applicazione.

1.5.3 Accesso a un'applicazione BSP.

Come spiegato in precedenza, si accedi alle applicazioni BSP utilizzando il protocollo HTTP o HTTPS (*Secure* HTTP). L'ambiente di *runtime* BSP utilizza la struttura HTTP di *SAP Web Application Server*. In *Internet communication Framework* (ICF), affinché un'applicazione BSP funzioni senza problemi è necessario un nodo, ovvero un servizio nella struttura dei servizi.

Un'applicazione BSP viene indirizzata ed eseguita tramite HTTP utilizzando un URL (Uniform Resource Locator). L'URL di un'applicazione BSP ha la seguente struttura (configurazione predefinita) come dimostrata nella figura 1.3:

<Prot>: // <Host>. <Dominio>. <Estensione>: <Porta> / sap / bc / bsp / <spazio dei nomi> / <nome applicazione>.

Protocol Prot è httpor https (se configurato).

- Host è il nome del server delle applicazioni che deve eseguire l'applicazione.
- Il dominio con l'estensione comprende diversi host con un nome comune. Questi possono essere parte di un singolo host o di una rete.
- Il numero di porta può essere omesso se viene utilizzata la porta predefinita 80 (http) o 443 (https). Per SAP *Web Application Server*, il numero di porta predefinito è 1080 (HTTP) o 1443 (HTTPS).
- Lo spazio dei nomi è l'ID del *namespace* dell'applicazione BSP.

- Le applicazioni SAP vengono fornite nello spazio dei nomi di SAP. Se si crea l'applicazione BSP in uno spazio dei nomi separato, deve essere utilizzato nell'URL. Il nome dell'applicazione è il nome dell'applicazione BSP come definito nell'ambiente di sviluppo.

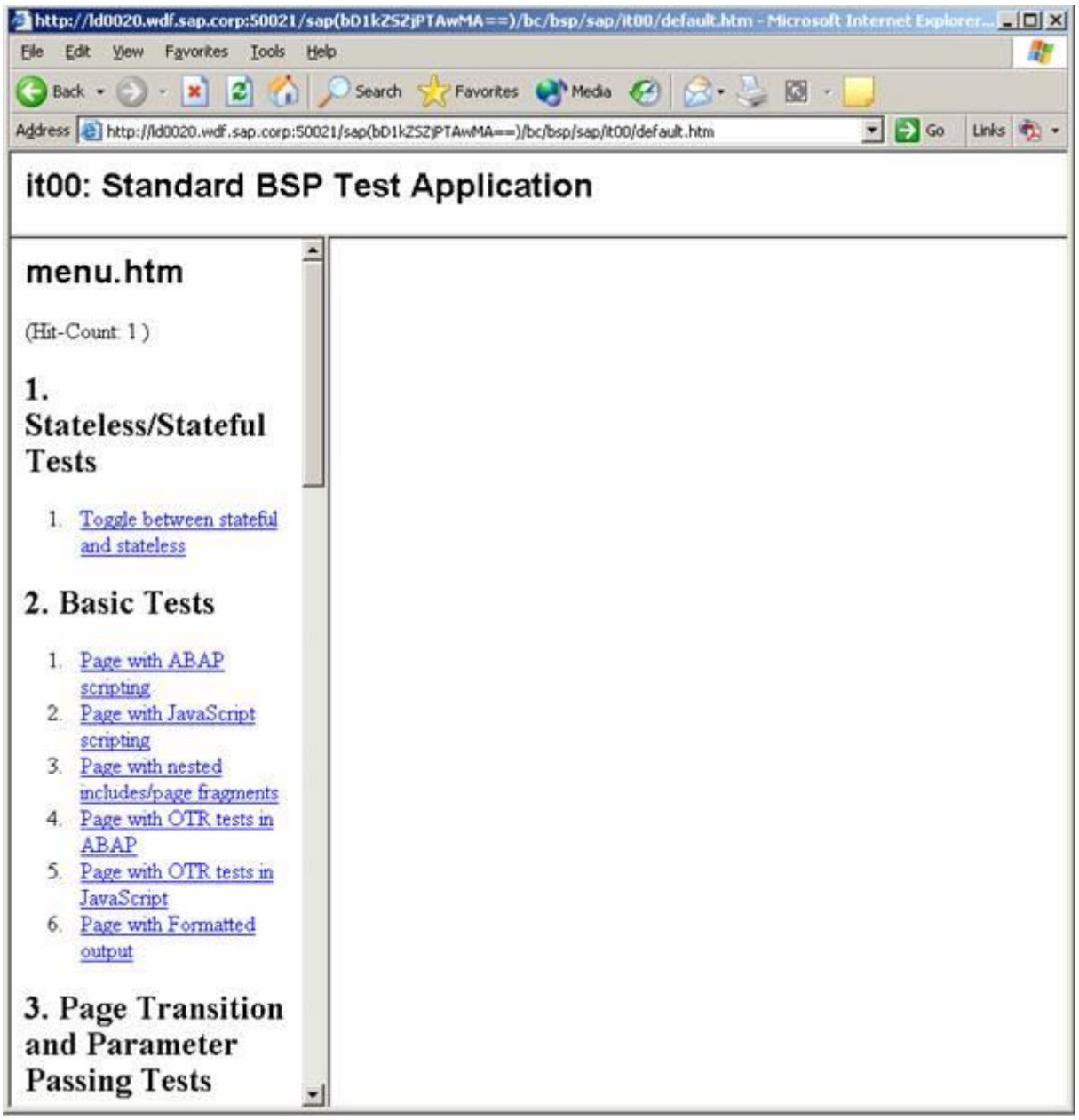

Figura 1.3 un esempio di una pagina: default.htm di un'applicazione BSP.

Se ciò non viene fatto, l'ambiente di runtime BSP restituisce un errore. È necessario inserire il dominio completo, in quanto potrebbero esserci problemi con i cookie HTTP (come Single Sign-On).

2.1 Elaborazione di un'applicazione BSP

Quando viene elaborata un'applicazione BSP, vengono eseguiti i singoli BSP e i relativi componenti. I corrispondenti passaggi di elaborazione vengono eseguiti in base al codice.
Se una richiesta HTTP viene inviata a un BSP, vale a dire, se viene chiamata una pagina (nel caso stateless: per la prima volta), la pagina viene istanziata in OnCreate. Successivamente, OnRequest viene chiamato. Viene quindi eseguito il gestore più appropriato per la richiesta, ovvero OnInitialization per l'input dell'URL, OnInputProcessing per l'input HTML o un altro gestore di eventi individuale se vengono utilizzati i controlli.
Se non lo ha già fatto, OnInitialization viene eseguito, in modo che si valutino i dati della fase di inizializzazione. In questa fase è possibile accedere ai dati nel sistema SAP, come il contenuto della tabella o BAPI. Quindi, viene elaborata la parte di layout (la parte nascosta del gestore eventi OnLayout), che determina la logica di progettazione e presentazione di una pagina. Questo codice è costituito da parti statiche (ad esempio HTML) e codice di script. Per garantire una programmazione pulita, nessuna logica aziendale o il recupero dei dati viene eseguito nella parte di layout. La parte di manipolazione diventa importante se il flusso di dati HTTP deve essere modificato in seguito. Il codice di manipolazione non dovrebbe contenere alcuna logica aziendale o istruzioni per la funzionalità di recupero dei dati.
Se non ci sono modifiche successive, la parte di manipolazione non è richiesta. Sulla base di tutti questi dati, la prima pagina viene creata e inviata all'utente.
Segue la fase di interazione dell'utente, in cui l'utente immette i dati. L'input dell'utente viene inviato nuovamente al BSP. Questo input attiva una richiesta HTTP aggiuntiva.
È possibile specificare per ogni pagina se è richiesta la modalità *stateful* o *stateless*. Questo viene fatto nella parte di elaborazione dell'input.
Ad esempio, l'input non corretto può essere controllato, i dati possono essere letti dal database e la pagina successiva può essere determinata.

La pagina successiva per la richiesta di navigazione è determinata nella parte di navigazione dell'applicazione. Se non viene impostata alcuna pagina successiva, la prima pagina viene rielaborata. Quindi, l'elaborazione della parte di inizializzazione continua.

2.2 Control Flow di un BSP.

Non è necessario utilizzare tutti i gestori di eventi mostrato nella figura 2.1, ad esempio un BSP può consistere solo del layout o solo contiene il layout, OnInitialization e OnInputProcessing.
Poiché il gestore di eventi OnDestroy è usato raramente, qui è irrilevante.

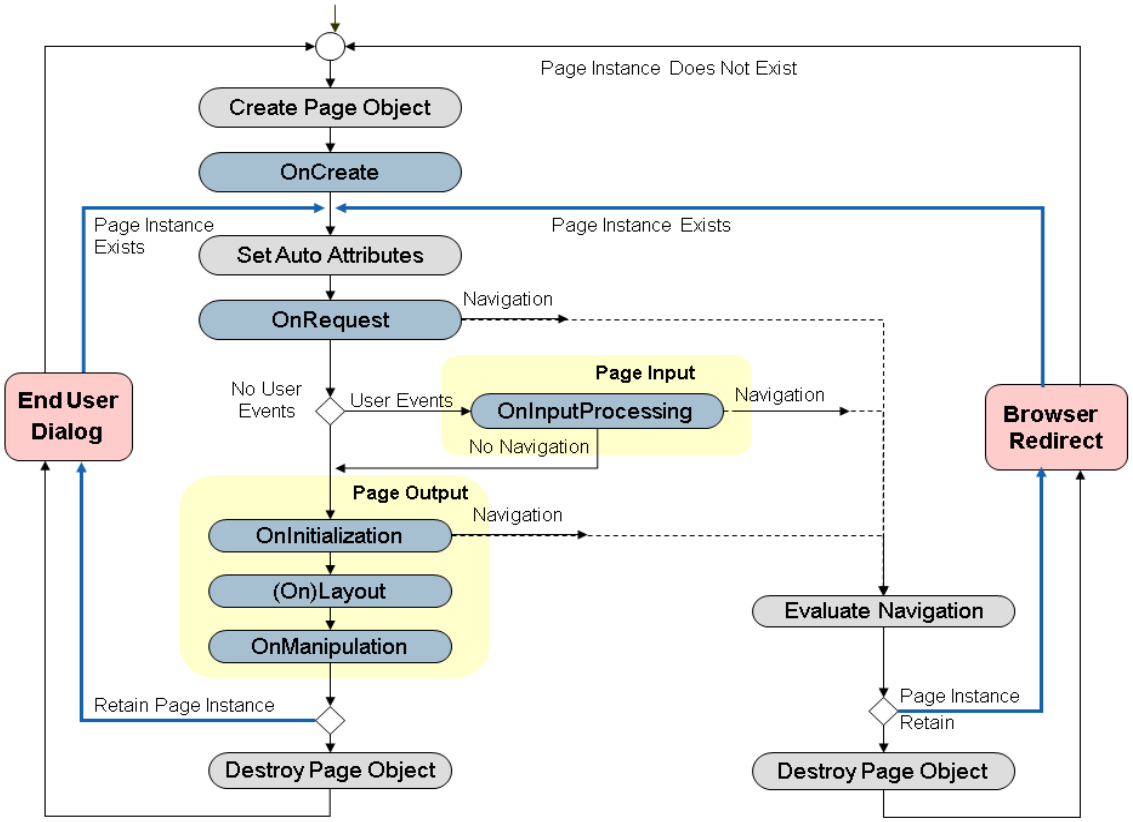

Figura 2.1: Il flusso di controllo di una pagina

Si distinguono i seguenti casi:

- BSP stateless
- BSP stateful

2.2.1 BSP Stateless.

Nel caso *stateless*, il contesto viene creato ogni volta che viene ricevuta una richiesta. Il contesto viene sempre distrutto quando viene inviata la risposta.

Per il *runtime* BSP, stateless significa: runtime-> keep_context = 0

Quando si crea l'output di una pagina, l'elaborazione è la stessa come nella figura 2.2.

Output di un BSP (Stateless)

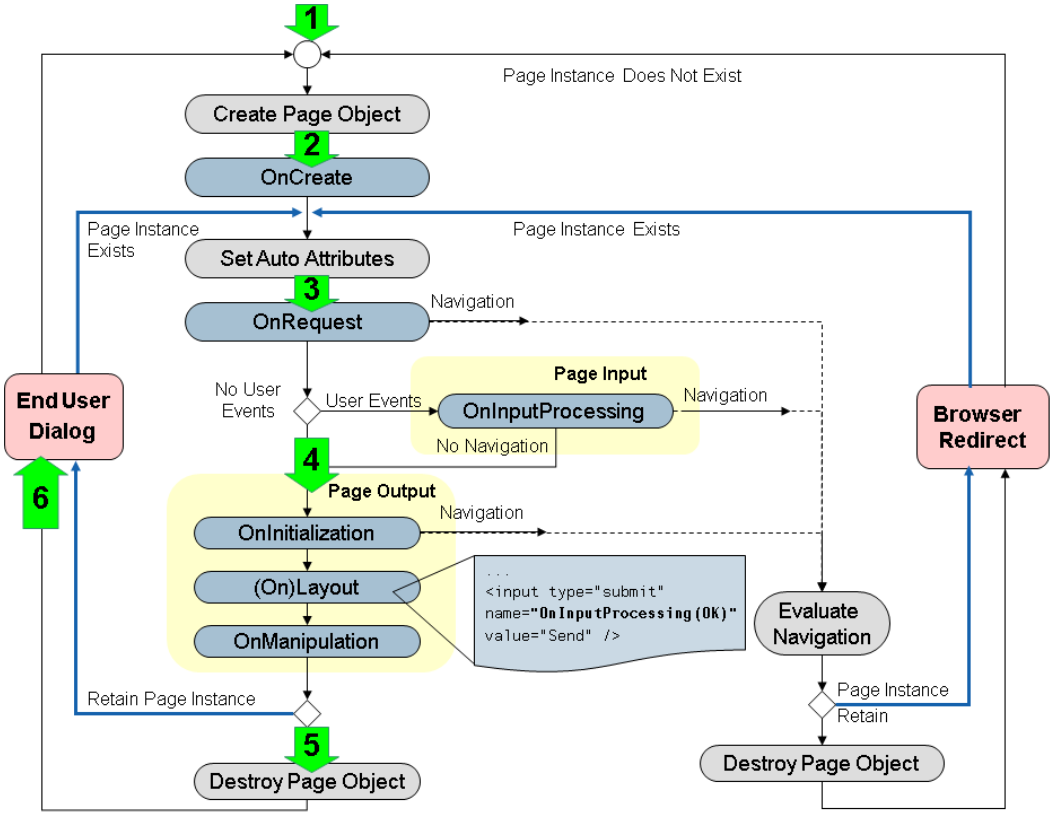

Figura 2.2: Il flusso di controllo di una pagina BSP Statless.

Quando si crea l'input per una pagina, l'elaborazione è la stessa descritta nella figura 2.3.

Input per un BSP (Stateless)

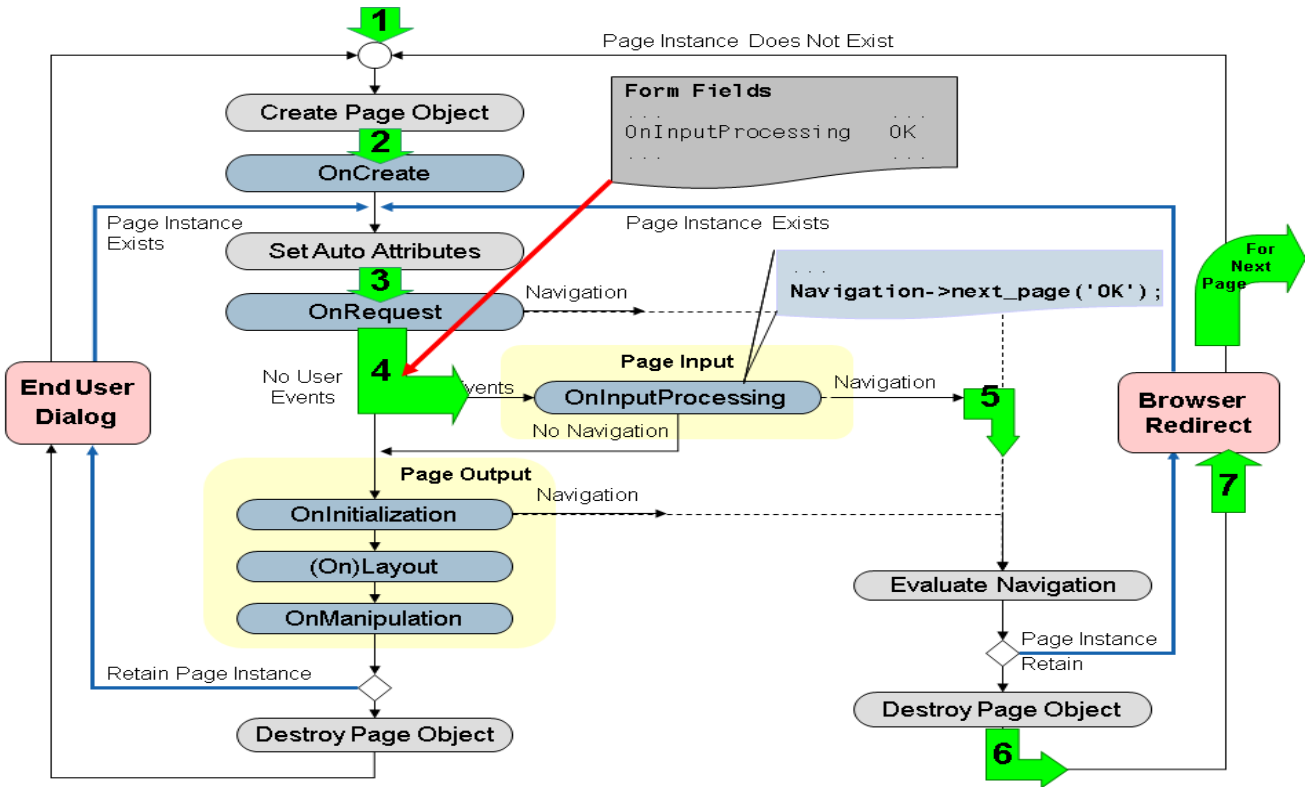

Figura 2.3: Input per un BSP (Stateless)

Con BSP Stateless (il valore predefinito è stateless), la navigazione può essere eseguita sulla stessa pagina o su altre pagine.

- **Navigazione all'interno del BSP.**
  Quando viene chiamata la pagina, il sistema verifica prima di tutto se esiste già un oggetto pagina per questo BSP. Se questo non è il caso, viene eseguito il gestore di eventi OnCreate, che crea l'oggetto pagina o un'istanza di esso. Quindi segue il gestore eventi OnRequest. Se esiste già un oggetto di pagina (in generale, questo non accade con la prima chiamata, proprio come nei BSP senza stato), il sistema si dirama direttamente sul gestore di eventi OnRequest. Quindi segue OnInitialization, se non esiste un interazione con l'utente e OnInputProcessing se c'è interazione con l'utente. Da OnInputProcessing, non vi è ulteriore navigazione in OnInitialization. L'elaborazione continua con il layout e quindi eventualmente con OnManipulation, sebbene questo gestore di eventi non venga utilizzato spesso. La pagina richiesta viene quindi visualizzata nel

browser. L'azione dell'utente può ora aver luogo, e quindi l'elaborazione ricomincia.

- **Navigazione verso altri BSP.**
  Il processo di elaborazione è molto simile a quando si naviga all'interno di un BSP. Se ora si passa da OnRequest a una pagina diversa, viene chiamato e valutato il gestore eventi *OnInputProcessing*. Segue *OnInitialization*, quindi il layout e così via.

2.2.2 BSP Stateful.

Nel caso di BSP *stateful*, il contesto è tenuto da una richiesta passata alla successiva. Per *il runtime* BSP *stateful* state: *runtime-> keep_context = 1*. Quando si crea l'output di una pagina, l'elaborazione è la stessa descritta nella figura 2.4.

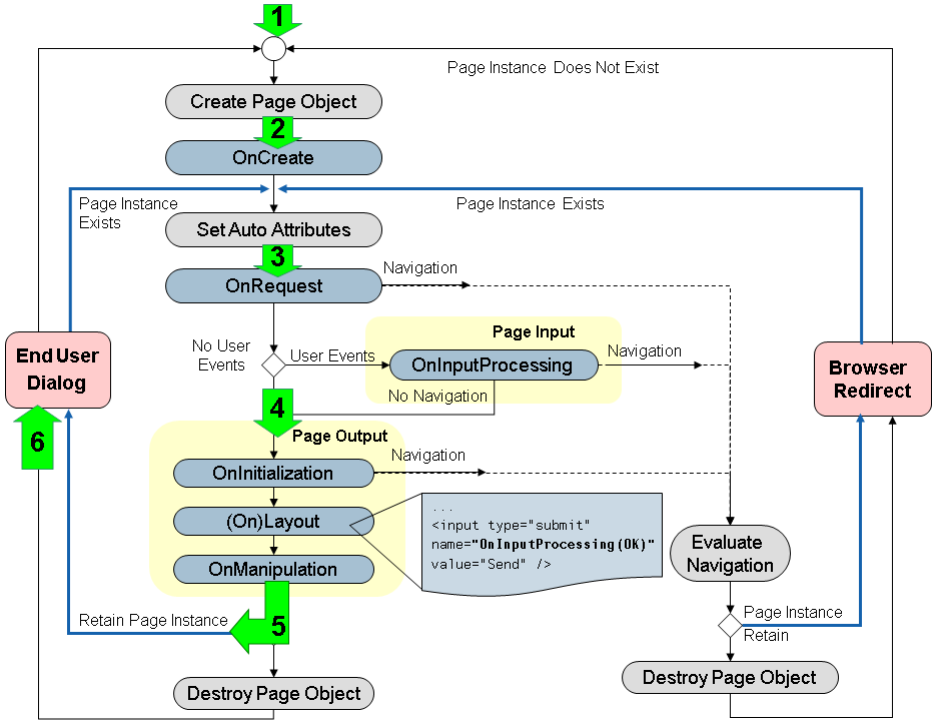

Figura 2.4: Output di un BSP (Stateful).

Quando si crea l'input per una pagina, l'elaborazione è la stessa nella figura 2.5.

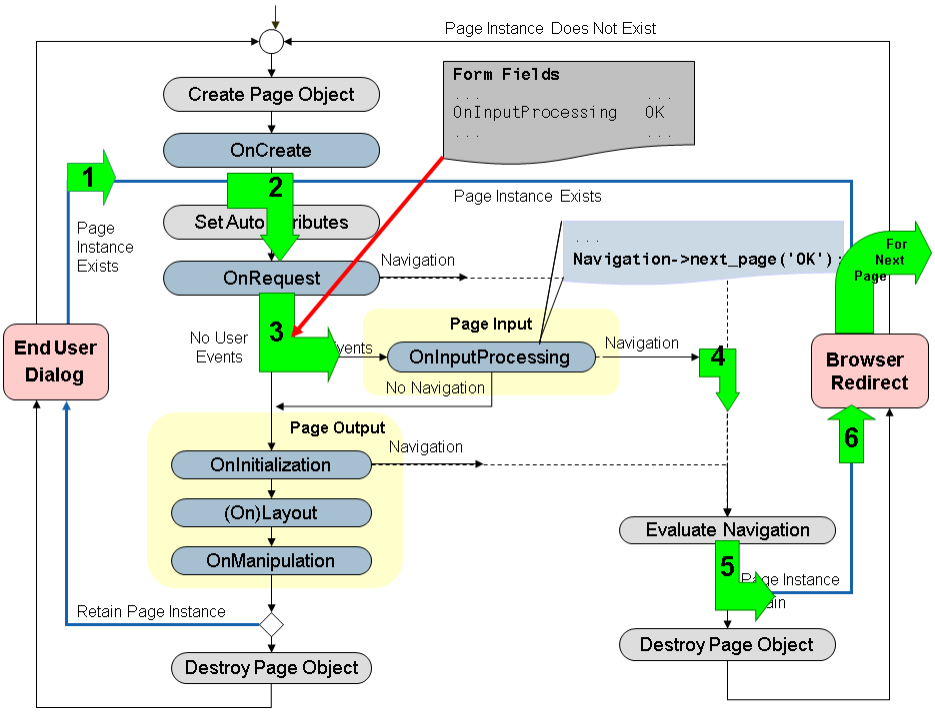

Figura 2.5: Input per un BSP (Stateful).

2.3 Creazione di un'applicazione BSP.

È possibile implementare il layout per ciascuna pagina (spesso chiamato BSP) utilizzando i linguaggi di script ABAP e JavaScript.
I BSP possono avere attributi diversi: pertanto, quando crei una pagina, la distinguiamo formalmente tra 3 tipi di pagina:

2.3.1 Pagina con logica di flusso.

Una pagina definita in questo modo può implementare l'intera logica di flusso stessa (modello basato su gestore eventi). La pagina può avere il gestore di eventi e i parametri di pagina (inclusi gli attributi automatici) oltre al layout.
Tale pagina è eseguibile e può essere indirizzata tramite un URL o tramite la navigazione da altre pagine.

2.3.2 VIEW

Le viste sono utilizzate esclusivamente per la visualizzazione dei dati dell'applicazione. Hanno parametri di pagina oltre alla sezione di layout. A differenza della pagina con logica di flusso, le viste non hanno né gestori di eventi né attributi di pagina automatici.
Una classe controller è generalmente assegnata a una vista. Questa classe controlla le chiamate di visualizzazione ed è responsabile della comunicazione con il modello.
Le viste sono basate sul modello di programmazione MVC e consentono di separare chiaramente la logica dell'applicazione dalla logica di presentazione nelle applicazioni BSP.

2.3.3 Frammento di pagina" XE.

I frammenti di pagina sono parti speciali. Non è possibile utilizzarli come pagine autonome in un'applicazione. Non sono eseguibili direttamente. Invece, li includi in altre pagine usando la direttiva include.
Esempio
<% @ include file = '/fragment.htm'%>.

2.4 Procedura

Per creare una pagina per la tua applicazione BSP dall'elenco degli oggetti (SE80):
1. Fare clic con il tasto destro del mouse sull'applicazione BSP.

2. Dal menu di scelta rapida, selezionare Crea → Pagina.
Il sistema visualizza la finestra di dialogo Crea pagina.

3. Fornire ulteriori specifiche sulla pagina inserendo il nome della pagina e una descrizione della pagina breve. Successivamente, selezionare l'opzione richiesta per il tipo di pagina (vista / pagina con logica di flusso / frammento di pagina).

4. Scegliere Questo grafico è spiegato nel testo di accompagnamento. Invio per confermare le voci. Il sistema visualizza l'editor di layout per la nuova pagina nell'area degli strumenti.

5. Scegli Questo grafico è spiegato nel testo di accompagnamento salva la pagina.

2.5 Creare un controller” XE.

È possibile creare un controller per utilizzare un modello di progettazione di Model View Controller nell'applicazione BSP. È possibile utilizzare un controller per l'accesso iniziale.

2.5.1 Procedura” XE

Utilizzare il Generatore di applicazioni Web in ABAP Workbench (*Transaction SE80*) per creare un oggetto controller come sub-oggetto della propria applicazione BSP.
In tal modo, si determina il nome del controller e il nome della classe del controller. L'URL viene automaticamente inserito dal nome dell'applicazione BSP e dal nome del controller.
Nel Class Builder (*Transaction SE24*), creare la classe derivata da CL_BSP_CONTROLLER2.
Salva e attiva la tua classe e il tuo controller.

2.6 Specifici dei parametri URL di sistema” XE

Per controllare l'ambiente di *runtime* BSP e l'applicazione BSP, è possibile utilizzare i seguenti parametri della stringa di *query* URL, in un URL di richiesta.

URL Parameter	Valore	Descrizione
sap-sessioncmd	Open: Close:	Riavvia l'applicazione BSP denominata nell'URL. Termina l'applicazione BSP specificata nell'URL e invia il browser, se specificato, al sito Web definito da sap-exiturl. Se si desidera che l'utente sia in grado di terminare l'applicazione, questa opzione dovrebbe essere disponibile in un'applicazione BSP come parte dell'interfaccia utente (come pulsante

		o collegamento ipertestuale).
sap-exiturl		Specifica l'URL a cui viene indirizzato il browser quando l'applicazione BSP termina (ad esempio, http://www.sap.com)
sap-theme		Sostituisce il tema per un'applicazione BSP per utilizzare altri oggetti MIME, come immagini e fogli di stile per la visualizzazione.
sap-themeRoot		Utilizzato per gestire i percorsi verso altre posizioni da cui vengono presi i fogli di stile.
sap-client		Specifica il client per un accesso a SAP Web Application Server; può essere utilizzato nell'URL o nei moduli HTML per sovrascrivere il client predefinito.
sap-user		Specifica l'utente per un accesso a SAP Web Application Server; può essere utilizzato nell'URL o in HTML per i moduli.
sap-password		Specifica la password per l'accesso; può essere usato nei moduli HTML.
sap-language		Specifica la lingua (ID lingua ISO, come EN o DE) per un accesso a SAP Web Application Server; può essere utilizzato nell'URL o nei

		moduli HTML per sovrascrivere la lingua di accesso predefinita.
sap-html-design		Cambio dinamico delle diverse varianti di progetto predefinite, senza utilizzare la progettazione dell'attributo di <htmlb: content>. I valori possibili sono CLASSIC, DESIGN2002 e DESIGN2003. Per SAP Enterprise Portal 5, è supportato DESIGN2002. Per SAP Enterprise Portal 6 sono supportati sia DESIGN2002 che DESIGN2003.
sap-domainRelax	off | min | max	Rilassamento massimo del dominio.
sap-accessibility	X |	Specifica per l'accessibilità Vedi anche Impostazioni di configurazione Per attivare questo indicatore, una casella di controllo attivata viene posizionata sulla schermata di accesso per l'aiuto di input.
sap-rtl	True | False	Dettagli del supporto RTL La famiglia di librerie <HTMLB> supporta il rendering RTL (da destra a sinistra) per design2003. È possibile attivare il rendering RTL utilizzando il parametro URL sap-rtl = true (ma solo per la prima chiamata prima che l'URL BSP venga rilevato sul mangling dell'URL). Usa l'attributo rtlAutoSwitch da <htmlb: content> per attivare automaticamente la modalità RTL, a

		seconda della lingua correntemente attiva (SY-LANGU). La decisione sull'attivazione di RTL è delegata al modulo funzione NLS_RTL_DISPLAY.

2.7 Classi di applicazione di un BSP Application

Un'applicazione BSP comprende un numero di oggetti di sviluppo diversi. Uno di questi oggetti è la classe dell'applicazione BSP.
La classe dell'applicazione, è una normale classe di oggetti ABAP. In quanto tale, la classe dell'applicazione può includere qualsiasi metodo, attributo ed evento che gli sviluppatori desiderano avere. La classe dell'applicazione viene generalmente utilizzata per archiviare dati e renderli disponibili su pagine BSP. Questi dati sono memorizzati come attributi. La classe dell'applicazione incapsula anche la logica dell'applicazione BSP nei metodi. Ciò consente a diverse applicazioni BSP di utilizzare la stessa classe di applicazioni e di fornire un'unica applicazione aziendale che contiene interfacce diverse, ad esempio per vari dispositivi, senza dover replicare la logica aziendale o dell'applicazione. Ciò significa anche che l'applicazione *Object globale* può essere utilizzata nell'applicazione BSP per accedere agli attributi e ai metodi della classe dell'applicazione.
Non devi usare una classe di applicazione nella tua applicazione BSP. Piuttosto, è un modo opzionale per gli sviluppatori di applicazioni di strutturare la loro applicazione BSP.
Si utilizza Web Application Builder nella transazione SE80 per assegnare una classe di applicazione a un'applicazione BSP.

2.7.1 Runtime Behaviour" XE

Qualsiasi classe di oggetti ABAP può potenzialmente essere utilizzata come una classe di applicazione di un'applicazione BSP. Tuttavia, l'ambiente di runtime BSP deve considerare la classe come un *singleton*, ovvero una classe per la quale esiste una sola istanza per sessione.
La durata di una classe di applicazione cambia a seconda che si tratta di

un'applicazione BSP stateful o stateless.
Nelle applicazioni BSP stateful, l'unica istanza della classe dell'applicazione, l'oggetto applicazione, viene generata alla prima richiesta inviata all'applicazione BSP. L'oggetto è quindi disponibile per l'intera durata della sessione. La durata della fine dell'oggetto dell'applicazione termina quando termina la sessione.
In modalità stateful, la classe dell'applicazione fornisce il buffering locale per set di dati difficili da determinare.

Nelle applicazioni BSP stateless, il contesto dell'applicazione (area di rollup) è disponibile solo per la durata di una singola richiesta e viene rilasciato alla fine della richiesta. Quando viene rilasciato il contesto dell'applicazione, vengono rilasciati anche tutti i dati e gli oggetti trattenuti dalla sessione sul server delle applicazioni. Questo include l'oggetto dell'applicazione. Ciò significa che la durata dell'oggetto dell'applicazione inizia quando la richiesta viene ricevuta e termina quando viene inviata una risposta. L'oggetto dell'applicazione non è disponibile su più pagine. Ogni pagina e ogni richiesta interagisce con una diversa istanza della classe dell'applicazione.
In modalità stateless, l'oggetto dell'applicazione non può contenere i dati tra le richieste. Per le applicazioni stateless, le classi di applicazione vengono solitamente utilizzate per archiviare la logica aziendale in metodi, ma non bufferizzano i dati.

2.7.2 Accesso all'oggetto dell'applicazione

Per accedere all'oggetto applicazione, si utilizza un riferimento oggetto digitato, che è archiviato come applicazione parametro in tutti i gestori di eventi di un BSP. Naturalmente, è necessario accertarsi che il parametro esista solo se per un'applicazione BSP è stata definita una classe di applicazione.

2.7.3 Requisiti per una classe di applicazione

L'unico vero requisito per un'applicazione è che il costruttore deve essere senza parametri. Altrimenti, la classe dell'applicazione non può essere istanziata dall’ambiente di runtime BSP.
Oltre a questo, non ci sono altre restrizioni. È necessario assicurarsi che l'implementazione interna dei metodi sia stata scelta correttamente,

a seconda della modalità di stato in cui è implementata la classe. Per le applicazioni stateless, ad esempio, sarebbe inutile implementare costose

routine di raccolta dati poiché queste andrebbero perse dopo ogni richiesta. Invece, ottieni solo i dati esatti di cui hai bisogno in quel momento. Nelle applicazioni stateful, è possibile implementare una fase di inizializzazione in cui si ottengono una grande quantità di dati contemporaneamente, il che può migliorare le prestazioni.
Nelle applicazioni BSP stateless, un'applicazione richiede spesso un controllo centralizzato in determinati momenti della durata dell'applicazione. Il modello BSP fornisce questa funzione per la classe dell'applicazione nell'interfaccia predefinita IF_BSP_APPLICATION_EVENTS.
Quando una classe di applicazione implementa l'interfaccia opzionale IF_BSP_APPLICATION_EVENTS, l'ambiente di runtime BSP richiama i metodi di interfaccia nei momenti rilevanti.

Quanto segue descrive i metodi e il tempo:

IF_BSP_APPLICATION_EVENTS ~ ON_START	Questo metodo viene chiamato dall'ambiente di runtime BSP quando l'applicazione BSP corrispondente viene avviata all'inizio della sessione BSP. Questo vale sia per applicazioni stateless che stateful. In genere, questo punto temporale viene utilizzato per eseguire controlli di autorizzazione che si applicano all'intera applicazione o per il recupero preliminare dei dati (in applicazioni con stato).
IF_BSP_APPLICATION_EVENTS ~ ON_STOP	Questo metodo viene chiamato dall'ambiente di runtime BSP quando l'applicazione BSP corrispondente viene terminata in modo esplicito. Questo vale sia per applicazioni stateless che stateful. Non è disponibile dopo ogni richiesta in applicazioni BSP

	stateless. Questa volta non viene valutata se la sessione viene terminata implicitamente da un timeout. In questo metodo, è solo possibile quindi eseguire operazioni opzionali non critiche. In genere, questo è un buon momento per le operazioni di pulizia come l'eliminazione dei cookie del browser o dei cookie sul lato server, se l'applicazione li ha generati.
IF_BSP_APPLICATION_EVENTS ~ ON_REQUEST	Questo metodo viene chiamato dall'ambiente di runtime BSP per ogni richiesta in arrivo a un BSP prima che venga dato il controllo a BSP (nel gestore di eventi OnRequest). Questa volta può essere utilizzata dalla classe dell'applicazione, ad esempio per ripristinare gli attributi che sono stati salvati in cookie client o lato server in una richiesta precedente.
IF_BSP_APPLICATION_EVENTS ~ ON_RESPONSE	Questo metodo viene chiamato dal runtime BSP per ogni risposta in uscita di un BSP dopo che il BSP è stato elaborato (dopo il gestore di eventi OnManipulation). Questa volta può essere utilizzata da una classe di applicazione stateless per attività quali il salvataggio degli attributi nei cookie lato client o lato server.

Se una classe di applicazione non ha già una super classe, può essere derivata dalla classe base predefinita CL_BSP_APPLICATION. Questa

classe fornisce i metodi che sono normalmente richiesti da un'applicazione BSP per l'incorporamento in un ambiente Web. In questo modo è possibile chiamare o impostare le informazioni sull'applicazione BSP corrente (come il timeout della sessione, l'URL corrente dell'applicazione BSP, la modalità stato e così via). Poiché l'oggetto applicazione è un attributo dell'applicazione in ogni gestore di eventi BSP, i metodi della classe CL_BSP_APPLICATION sono anche disponibili con l'ereditarietà corrispondente. Ciò semplifica la fornitura della funzionalità legata ai livelli di applicazione inferiori utilizzando un riferimento a un singolo oggetto.

In questa capitolo, sarà illustrata la progettazione riguardante il software presentato:
Verranno mostrati e spiegati in maniera opportuna i requisiti funzionali del sistema, e verranno inoltre descritte le specifiche del software più nel dettaglio, nonché i diagrammi UML utilizzati.

3.1 Specifiche del Software

Si progetta un applicazione BSP che permette di interfacciarsi con il gestionale aziendale (SAP) per gestire la produzione e la logistica. L'interfacciamento deve avvenire mediante un dispositivo mobile, nel nostro caso un palmare, che permette di gestire sia un magazzino attraverso il modulo WM (warehouse management) cioè per la gestione del magazzino che MM (material management) per la gestione delle materie prime del gestionale SAP. L'interfaccia deve essere così suddiviso:

- **Logistica**
  - **Ordini Cliente –** per gestire tutti gli ordine destinato ad un cliente.
  - **Gestione Magazzino**
    - Trasferimento merce – possibilità di spostare prodotti/materie prime da un area all'atra.
    - Carico da Acquisto- inserire nel sistema materiali acquistati.
    - Carico da Acquisto *Interplant* – si possono acquistare da altri impianti situati in citta diversi della stessa azienda.
    - Interrogazione – avere un visione generale del magazzini.
  - Inventario - effettuare l'inventario del magazzino.

- **Produzione**
  - Trasferimento Merce- possibilità di spostare prodotti/materie prime da un area all'altra.
  - Carico da Produzione - inserire nel sistema prodotti finiti.
  - Carico da Acquisto - inserire nel sistema materiali acquistati.
  - Approntamento per Produzione- preparazione della materia

prima per la produzione.

- Interrogazione – controllare la presenza o meno di un determinato prodotto/materie prime.
- Consumo materie prime – utilizzo di materie prime presente nel magazzino.
- Approntamento materie prime – elaborazione di materie prime.
- Inventario - effettuare l'inventario del magazzino.
- Approntamento MP x Area- – elaborazione di materie prime di un determinato area del magazzino.
- Rich. Manuale Materiale su Linea – richiedere il manuale di un prodotto.
- Conferma Fase (Resa / Scarto)
- Conferma Inizio / Fine Fase

- ***Logout***

Per poter avere un visione generale delle attività svolto, ogni utente si deve autenticare e dopo aver effettuato l'accesso, in base al id ad esso assegnato, accede solo all'area di interesse: gli utenti del settore di produzione posso accedere solo il meno di produzione e non possono accedere al meno di logistica ed viceversa. Salvo alcuni utente che posso accedere ad entrambi i menu.

3.2 Requisiti Funzionali e Non Funzionali

Verranno ora presentati i requisiti funzionali dell'applicativo, ossia come il sistema deve reagire agli input e comportarsi nelle varie situazioni e scenari possibili.
I requisiti funzionali possono essere raccolti in forma testuale, discorsiva. In questo caso si tratta di un metodo immediato e tradizionale ma non formale. Per rappresentare i requisiti è preferibile adottare un modo più o meno formale. I requisiti funzionali raccolti sono strutturati in schede aventi i seguenti campi:
Titolo: strutturato in tre sezioni:

- *Codice*: identifica il requisito.
- *Area di riferimento*: collocazione del requisito.
- *Titolo specifico:* descrive brevemente la funzionalità del requisito;

Input: sono le informazioni che sono inserite all'interno del processo come ingresso;
Descrizione: sequenza di azioni eseguite dall'operatore, eventuali messaggi di errore come risposta ad anomalie e parametri che incidono sull'output;
Output: risultato del processo.
Dalla Figura 3.1 alla Figura 3.7 vengono illustrati alcuni esempi di requisiti funzionali.

RF01	Accesso all'interfaccia
Input	Inserimento del *user*-id
Descrizione	L'utente inserisce il codice *user*-id nell'menu di login. Effettua l'accesso e assecondo che lavora nella logistica o nella produzione, accede al menu dedicata.
Output	Presentazione del menu con l'elenco dei voci e operazione che può effettuare.

Figura 3.1 Requisiti funzionale.

RF02	Trasferimento Merce
Input	Inserimento/ rilevamento del codice del materiale o prodotto. Inserimento dell'area di destinazione.
Descrizione	L'utente accede al menu del trasferimento, con il palmare rileva il code materiale, le si presenta un meno con le possibili destinazione del materiale che potrà selezionare.
Output	Materiale o prodotto non più visibile nella su detta area o *plant* ma presente nella destinazione scelta.

Figura 3.2 requisito funzionale.

RF03	Interrogazione
Input	Inserimento codice area
Descrizione	L'utente accede al menu del interrogazione, le si presente un menu con le are del magazzino che può visualizzare il loro contenuti.
Output	Visualizzazione di un elenco dei materiali presenti in tale area.

Figura 3.3 requisito funzionale

RF04	Approntamento MP x Area / Approntamento materie prime
Input	Inserimento codice area / codice della materia prima.
Descrizione	L'utente dal menu di approntamento, a seconda che sia per area o per un generi materie prima, seleziona l'area e materie prima che successivamente vera segnato come preparato per essere utilizzato.
Output	Visualizzazione della materia prima come preparato cioè pronto ad essere impiegato nella produzione.

Figura 3.4 requisito funzionale

RF05	Consumo materie prime.
Input	Selezione della materia prima precedentemente preparato.
Descrizione	Accedendo al menu di consumo materie, all'utente viene presentato un elenco dei materie prime pronto ad essere consumato.
Output	Codice materiale viene cancellato dalle materie prime.

Figura 3.5 requisito funzionale

RF06	Carico da Produzione
Input	Inserimento del codice del nuovo prodotto dalla produzione.
Descrizione	Un prodotto nuovo viene registrato disponibile in magazzino.
Output	Presenza del prodotto nel magazzino, quindi disponibile.

Figura 3.6 requisito funzionale

RF07	Carico da Acquisto / Carico da Acquisto *Interplant*
Input	Inserimento del codice del nuovo materiale/ prodotto acquistato da un cliente esterno oppure da un'altra magazzino.
Descrizione	Quando un materiale viene acquistato sia che si tratta di un acquisto interno della stessa azienda o dall’esterno viene registrato inserendoli nel sistema (gestione aziendale).
Output	Presenza del materiale nel magazzino, quindi disponibile.

Figura 3.7 requisito funzionale

3.3 Progettazione con UML

3.3.1 Attori del sistema

È opportuno, prima di passare alla descrizione dei diagrammi vera e propria, descrivere gli attori del sistema software progettato e i relativi scenari: per attore si intende una entità che deve interagire con il sistema software, e può essere una persona fisica o una astrazione generale (come ad esempio un altro software). Gli attori hanno un ruolo ben definito all'interno del sistema, e possono compiere dunque svariate azioni, che vengono denominate casi d'uso; questi sono una tecnica utilizzata per identificare i requisiti funzionali di un sistema software, descrivendo le azioni compiute dagli attori che interagiscono col sistema, valutandone le varie interazioni. Gli attori del software in questione sono:

• *Dipendente:* è la figura presente all0interno dell'azienda, quindi registra sul sistema tutte le transazione e tutte le attività che riguardano il magazzino.

3.3.2 Casi d'Uso

In primo luogo, verrà presentato il diagramma dei Casi d'Uso dell'applicazione. È un diagramma strutturale che modella e descrive il comportamento del sistema, ossia come il sistema agisce e reagisce e ciò che è visibile all'esterno. Non modella i suoi requisiti funzionali, ma illustra i vari scenari ai quali gli attori possono prendere parte, esprimendo un comportamento, offerto o desiderato, sulla base dei suoi risultati osservabili. Gli omini stilizzati rappresentano gli attori del sistema precedentemente descritti, i cerchi ovali i casi d'uso, il rettangolo contenente i casi d'uso rappresenta il sistema, le linee, le varie relazioni tra attori e casi d'uso. Per relazione si intende una interazione che può avvenire tra attori, sistema software e/o altri attori; in genere, l'attore rappresenta l'esterno del sistema, mentre i casi d'uso si trovano al suo interno. La relazione che lega l'attore con i diversi casi d'uso è un'associazione, una relazione che rappresenta una comunicazione tra attori e casi d'uso interni al sistema.
Nella Figura 1.9 è rappresentata con un semplice segmento che congiunge le due figure.

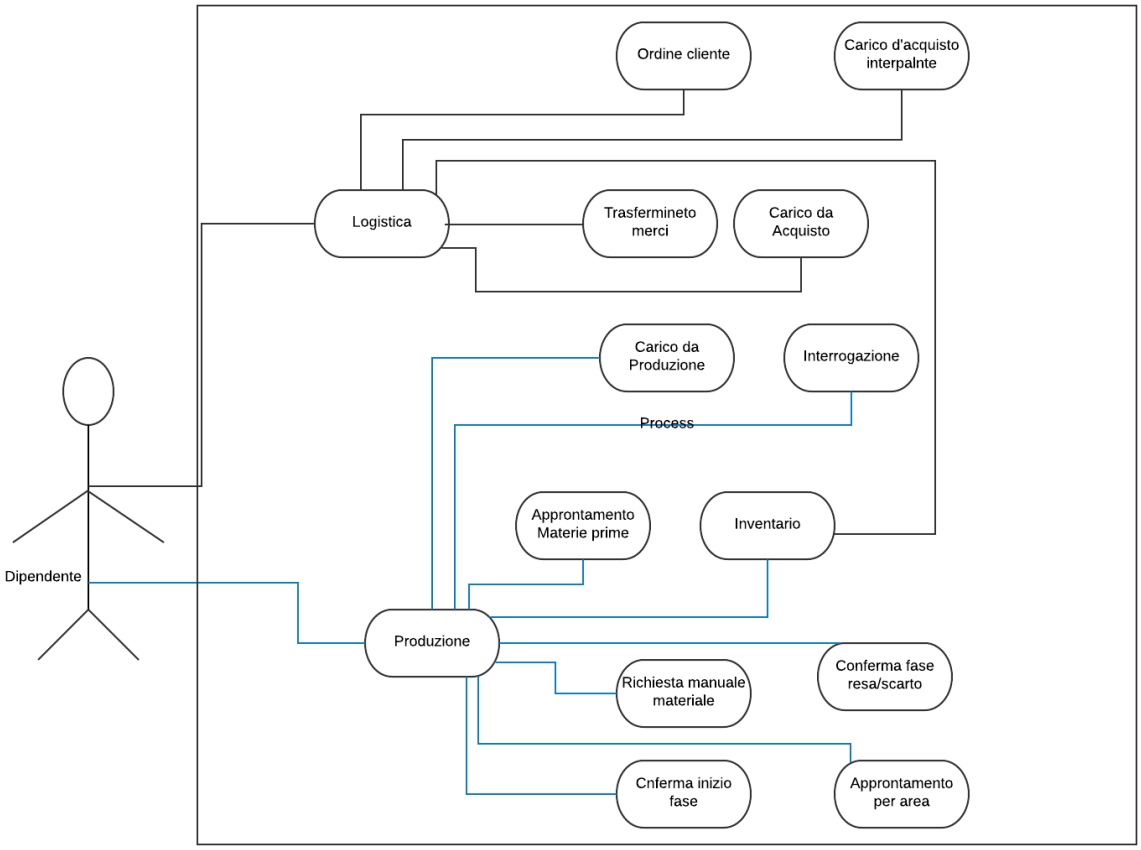
Ordine cliente
Carico d'acquisto interpalnte
Logistica
Trasfermineto merci
Carico da Acquisto
Carico da Produzione
Interrogazione
Process
Approntamento Materie prime
Inventario
Dipendente
Produzione
Conferma fase resa/scarto
Richiesta manuale materiale
Cnferma inizio fase
Approntamento per area

Figura 1.9

Si illustra parti di codici per i component di un applicazione BSP, in modo particolare il contenuto dei seguenti file mostrata nella figura 4.1.

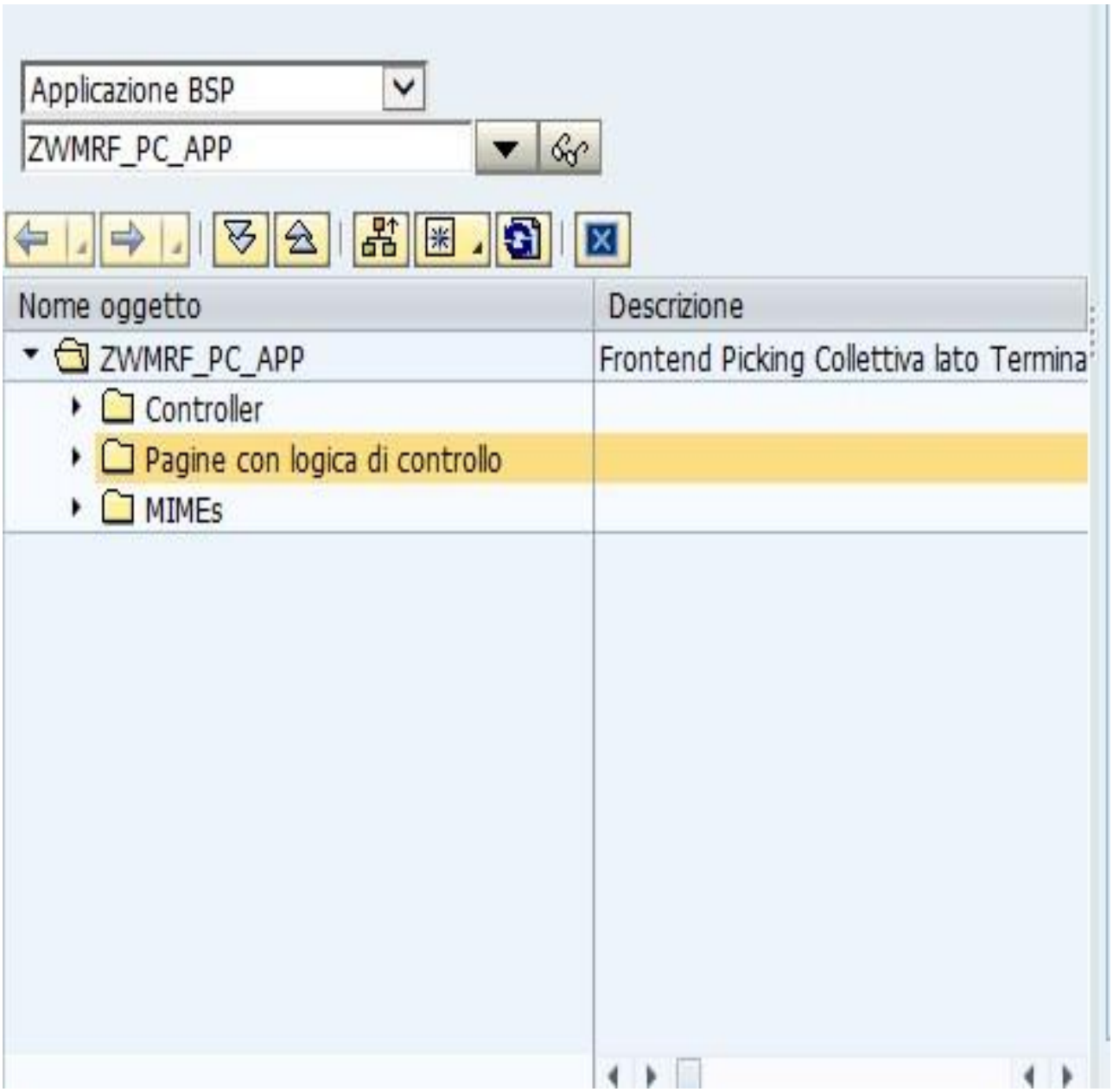

Figura 4.1: file contenuto in un applicazione BSP.

4.1. MIME

La cartella MIME del progetto BSP contiene le sotto cartelle:

- CSS
- Javascript
- Immagini: utilizzati nelle varie pagine html.

4.1.1 Cartella CSS

La cartella contiene la definizione del CSS (*Cascade Style Sheet*) che descrive come gli elementi nella pagina html si devono posizionare all'interno della pagina web. Come mostrano le figure 4.2, 4.2a, 4.2b.

```
html {
        overflow: hidden;
}/*
 * app width: 210px
 * list row (total) height: 30px
 */
/* Generic Styles */
#arrow_up {
        width: 210px;
        height: 9px;
        visibility: hidden;
        text-align: center;
        /*font-size: 8px;*/
        background-image: url("../images/arrow_up.gif");
        background-repeat: no-repeat;
  background-position: center;
  /*margin-bottom: 5px;*/
  /*padding: 5px 5px 10px 5px;*/
  padding: 0px 5px; 0px; 5px;
}
```

Figura 4.2: CSS per le pagine web.

```
#arrow_down {
        width: 210px;
        height: 9px;
        visibility: visible;
        text-align: center;
        /*font-size: 8px;*/
        background-image: url("../images/arrow_down.gif");
        background-repeat: no-repeat;
  background-position: center;
  /*margin-bottom: 5px;*/
        /*padding: 5px;*/
        padding: 0px 5px; 0px; 5px;
        position: relative;
        top: -5px; }
.list {
        font-family: Helvetica, Arial, sans-serif;
        font-size: 8px;
        color: #333;
        overflow: hidden;
}
```

Figura 4.2a: CSS per le pagine web.

```
#header {
        font-family: Helvetica, Arial, sans-serif;
        font-size: 8px;
        color: #333;
        background: #DFDFDF;
        height: 50px;
        padding: 5px;
        width: 210px; }
#header .generalInfo {
        height: 10px;
        border-bottom: 0px solid;
        padding: 1px 5px 2px 5px;
        background-color: #B2B2B2; }
#header .headerTitle {
        float: left;
        font-weight: bold;
        padding-top: 1px; }
```

Figura 4.2b: CSS per le pagine web

```
#header .headerUser {
        float: right;
        font-style: italic;
        padding-top: 1px;
}
#header .userIcon {
        float: right;
        width: 10px;
        height: 10px;
        background-image: url("../images/ic_user_small.gif");
        background-repeat: no-repeat;
  background-position: center;
  padding-bottom: 0px;
  padding-right: 5px; }
#header .pickInfo {
        padding-top: 5px;
        padding-bottom: 0px;
        border-bottom: 1px solid;
        height: 10px;
}
```

Figura 4.2c: CSS per le pagine web.

4.1.2 Javascript

La cartella *javascript* contiene l'implementazione di funzione *javascript* per gestire l'interazione dinamiche dell'utente.

In totale sono stati definiti tre file di javascript:

global.js – in cui sono definiti variabile globali.

Navigation.js – nella quale sono definiti le funzione che verranno chiamate a seguito di un azione compiuto da parte di un utente.

Commandbox.js –in cui si utilizza entrambi i file sopra menzionate

La figura 4.2 e 4.2a mostrano i codice *javascrip*t contenuto nel fil commandbox.js, utilizzate per gestire l'interazione che l'utente avrà con l'applicazione.

```
/*  * Requires global.js navigation.js
 * Controller for command box
 *  * <div class="command_box">
 *        <div id="icon"></div>
 *        <div id="message"></div>
 * </div>
 *  */
var messageFlag = false;
function clearCommandBox() {
        var message = document.getElementById('message');
        var icon = document.getElementById('icon');
        message.innerHTML = '';
        removeClass(message, 'error');
        removeClass(message, 'warning');
        removeClass(message, 'ok');
        removeClass(icon, 'error');
        removeClass(icon, 'warning');
        removeClass(icon, 'ok');
}
```

Figura 4.2: Javascript per la gestione delle pagine web.

```
/*
 * cls = class to be applied to the icon and message container (admitted values: error, warning, ok)
 */
function showMessage(cls, msg) {
	clearCommandBox();
	messageFlag = true;
	var message = document.getElementById('message');
	var icon = document.getElementById('icon');

	message.innerHTML = msg;
	addClass(message, cls);
	addClass(icon, cls);
}
```

Figura 4.2a: Javascript per la gestione delle pagine web.

4.1.3 Immagine

La figura 4.3 mostra in contenuto del file immagine.

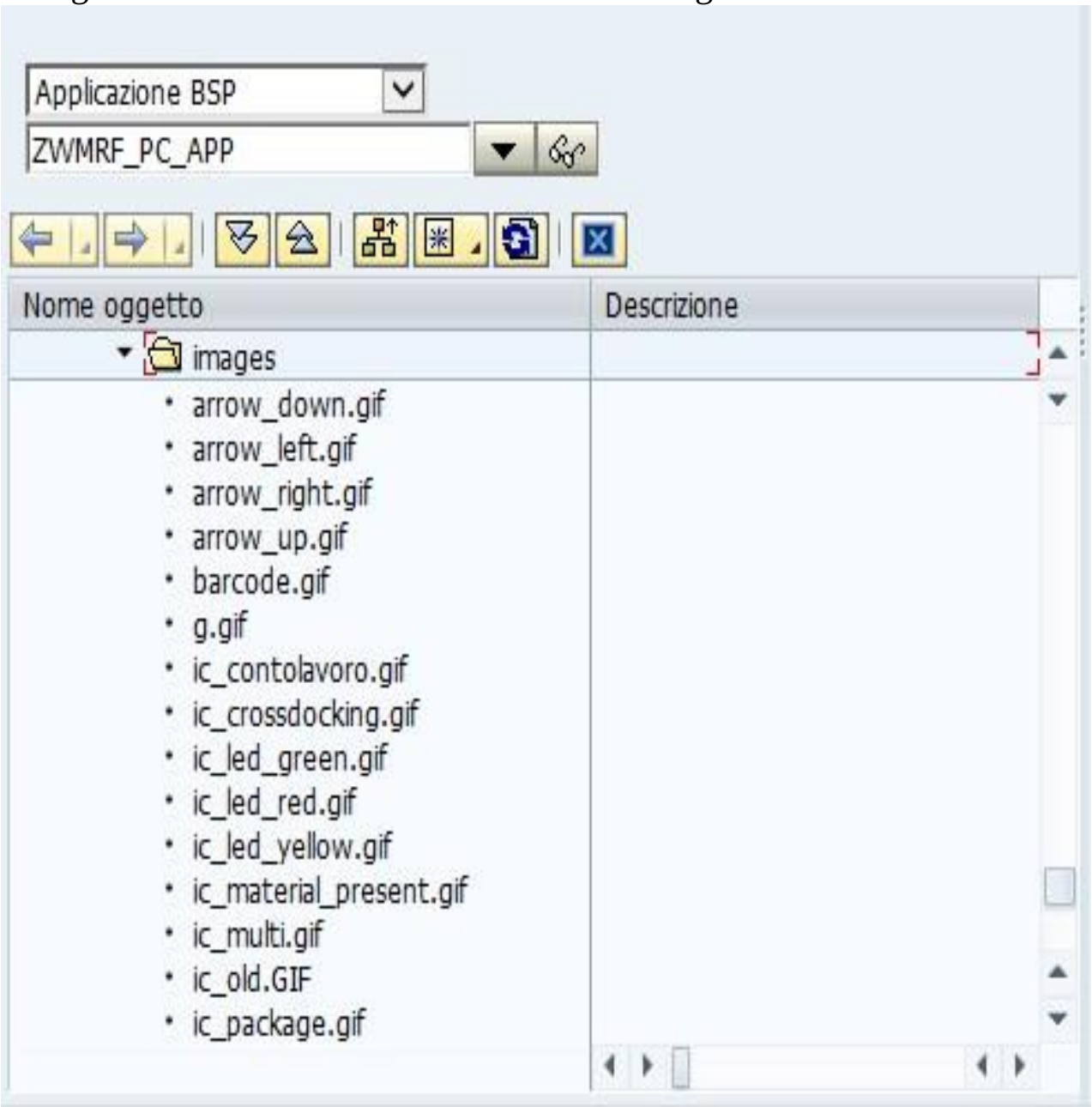

Figura 4.3: Immagine utilizzati per rendere la pagine web presentabili.

4.2 Controller.

Il folder controller detiene la logica aziendale e i dati dell'applicazione. Adatta per il *rendering* della risposta all’utente, come succede nella paradigma Model View Controller (MVC).

4.3 Pagine di controllo.

Le pagine di controllo il codice per la gestione dell’applicazione e sono suddivisa nelle seguente sottocartelle che contengono rispettivamente:

- Attributi della pagina
- *Layout* che contiene il file *html* della pagina
- *Event handler* che gestisce cosa deve accadere quando si apri la pagina nel browser.

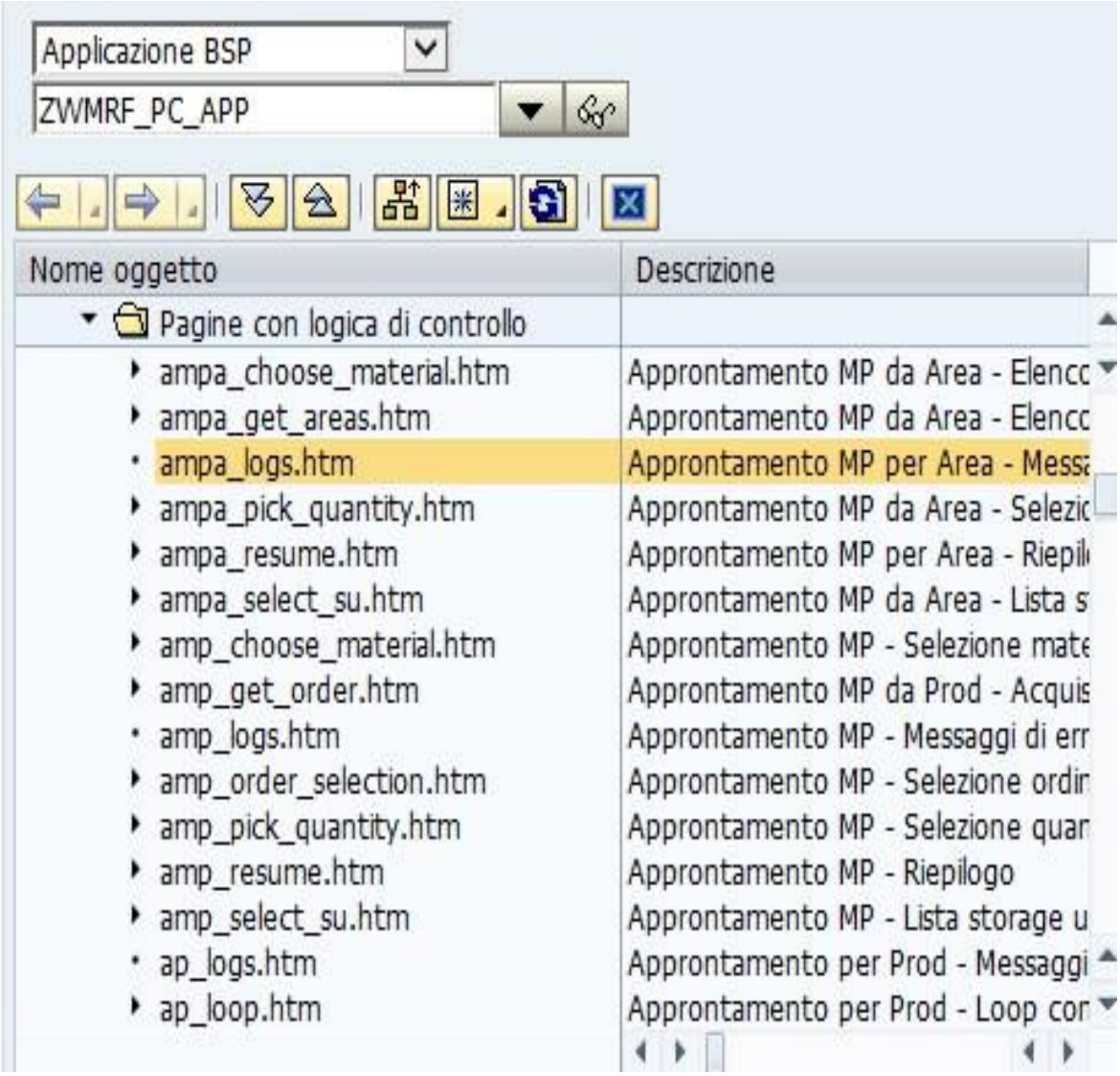

Figura 4.3: Contenuto del file logica di controllo di un BSP applicazione.

4.3.1 Attributi

Gli attributi sono dei oggetti dati dichiarato all'interno di una classe. Si definisce un attributo specificando il nome, la categoria, il tipo ed una descrizione come mostra nella figura 4.4. Un attributo può essere:

- Statico: sempre presente.
- Istanza: sempre assegnato a un oggetto di una classe e determina il suo stato.
- Privato: visibile solo all'interno della sua classe.

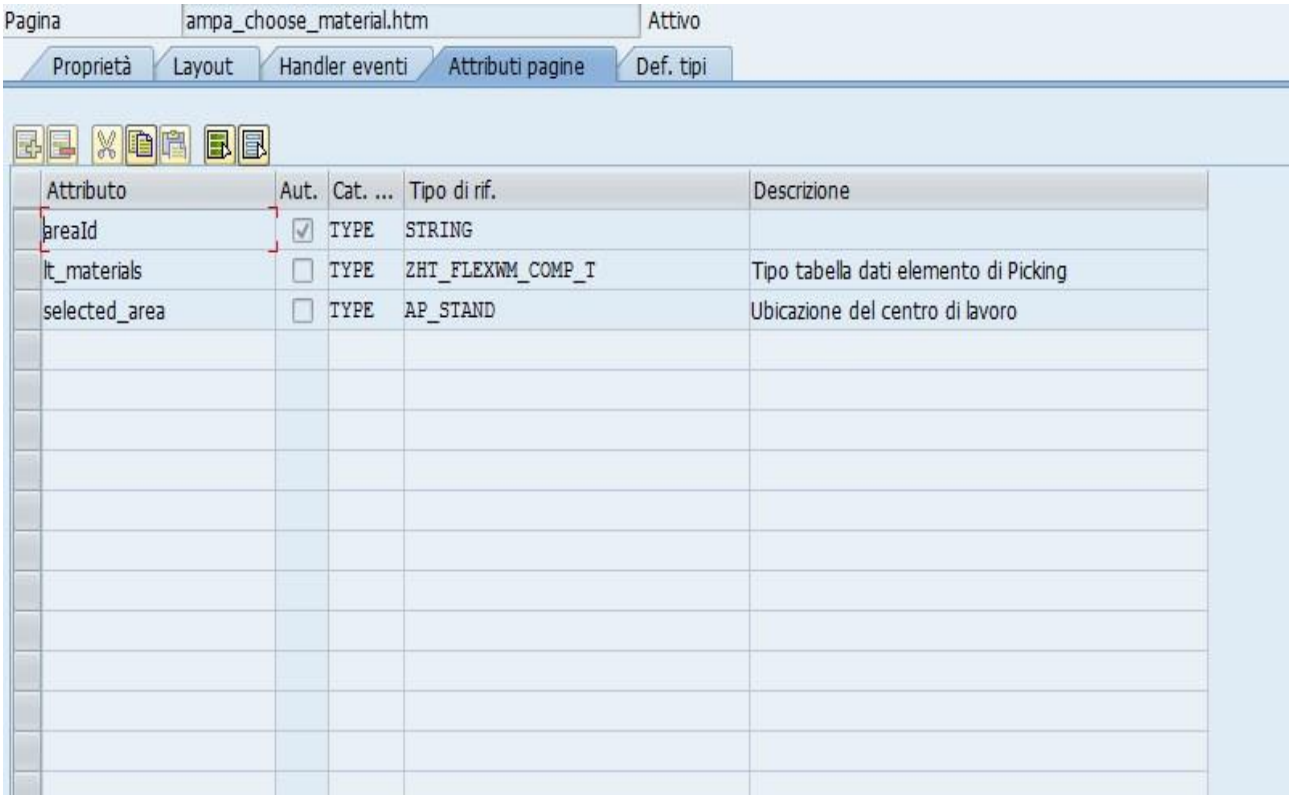

Figura 4.4: Definizione di tre attributi che definisce un materiale scelto dal magazzino.

4.3.2 Layout

Nei layout vengono definiti (i views) le pagine HTML perla visualizzazione degli eventi gestiti dall'event handler; la figura 4.5 a 4.5e mostrano una possibile definizione di un layout.

```
<%@page language="abap"%>
<%  DATA    str_tabindex    TYPE STRING.%>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
"http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <link rel="stylesheet" href="css/new_style.css">
  <script language="JavaScript" src="js/globals.js"></script>
  <script language="JavaScript" src="js/navigation.js"></script>
  <script language="JavaScript" src="js/commandBox.js"></script>
  <script language="JavaScript" src="js/utils.js"></script>
  <title>Pick</title>
</head>
<body>
<div class="box auth_header">
  <div class="generalInfo">
    <div class="title">Appr. MP Area</div>
    <div class="user"><%= application->USER %></div>
    <div class="userIcon"></div>
  </div>
  <div class="screenInfo">
    <div class="caption">Spara la SU dest.</div>
  </div>
  <div class="materialInfo">
    <div class="row1">
      <div class="matNr c_material"><%= application->header_data-matnr %></div>
      <%
      DATA: lv_tmp_str        type string.
      lv_tmp_str = application->header_data-maktx.
      "--> Trim text
```

Figura 4.5: Pagine html per la visualizzazione.

```
    zht_flexwm=>trim_text(
      exporting
        i_text    = lv_tmp_str
        i_type    = 'H'  " Type h or null
      importing
        o_text    = lv_tmp_str
    ).
    %>
    <div class="materialDesc c_material"><%= lv_tmp_str %></div>
   </div>
   <div class="row2">
      <div class="headerBatch c_batch"><%= application->header_data-charg %></div>

    <div class="qtyLeft"><%= application->header_data-order_num %></div>
   </div>
  </div>
</div>
<div class="spacer"></div>
<div class="blank_box pickInfo">
   <div class="left">
    <div class="desc">Bancale</div>
    <div class="data"><%= application->PC_ACTUAL_SU-LENUM_TEXT %></div>
   </div>
   <div class="right">
    <div class="desc">Ubicazione</div>
    <div class="data"><%= application->PC_ACTUAL_SU-LGPLA %></div>
   </div>

</div>

<!--<form id="pick_form" name="pick_form" action="ap_check_su.do" method="POST">-->
```

Figura 4.5b: Pagine html per la visualizzazione.

```
 CALL FUNCTION 'CONVERSION_EXIT_CUNIT_OUTPUT'
     EXPORTING
      INPUT = su_selection_item-PC_UM
      LANGUAGE = SY-LANGU
     IMPORTING
      OUTPUT = txt_meins.
  %>
   <div class="label2"><%= lv_roundup_qty %> <%= txt_meins %></div>
   <div class="label2"><%= su_selection_item-PC_PROP_QTY_TEXT %> <%= txt_meins
%> </div>
  </div>
  <%
   ENDLOOP.
  %>
</form>
<div class="spacer"></div>
<div class="box command_box">
          <div id="icon"></div>
          <div id="message"></div>
</div>
<script language="JavaScript">
 init();
 //document.getElementById('qty_1').select();
 var err = getURLParameter('err');
 switch (err) {
  case '1':                                                              em-
   showMessage('error','SU non trovata');
   break;
 }
 onLeftPressed = function(element) {
  moveToPage('ampa_select_su.htm');
  return true;
```

Figura 4.5c: Pagine html per la visualizzazione

```
  var commandBox = document.getElementById('message');
  var currentContent = commandBox.innerHTML;
  commandBox.innerHTML = currentContent + String.fromCharCode(e.keyCode);
};
onBackspacePressed = function(element) {
  clearCommandBox();  };
onEnterPressed_old = function(element) {
  var error = false;
  if (messageFlag) {
    clearCommandBox();
    messageFlag = false;
    document.getElementById('qty_1').select();
  } else {
    if(!isNumeric(document.getElementById('qty_<%= str_tabindex %>').value))
      error = true;
    if(!error && parseInt(document.getElementById('qty_<%= str_tabindex %>').value) == 0)
      error = true;
    if(!error && document.getElementById('qty_<%= str_tabindex %>').value.length == 0)
      error = true;
    if(document.getElementById('qty_<%= str_tabindex %>').value.length > 4)
      error = true;
    <% ENDLOOP. %>
    if(error) {
      showMessage('error','Quantit&agrave; errata');
    } else {
      clearCommandBox();
      submitForm('pick_form');   }    }
  return true;
};
onEndFocusSequence = function(formId) {
  //submitForm('pick_form');  };
onHomePressed = function(element) {
   moveToPage('menu.htm');
```

Figura 4.5d: Pagine html per la visualizzazione

```
onDownPressed = function(element) {
 if(!currentField)
  document.getElementById('qty_1').select();
 else
  focusNext('pick_form');
 return true;
};
onUpPressed = function(element) {
 //if(currentField)
 //  focusPrev('pick_form');
 var roundUpValue = <%= su_selection_item-ROUNDUP_QTY %>;
 document.getElementById('qty_1').value = roundUpValue;
 return true;
};
onEnterPressed = function(element) {
 if (messageFlag) {
  clearCommandBox();
  messageFlag = false;
  return true;
 }
 if (!document.getElementById('loading')) {
  var barcode = document.getElementById('message').innerHTML;
  document.getElementById('ean').value = barcode;
  document.getElementById('qty').value = document.getElementById('qty_1').value;
  doAction('getSU','su_form');
 }
 return true;
};
 onDigitPressed = function(e, element) {
 if (messageFlag) {
  clearCommandBox();
  messageFlag = false;    }
```

Figura 4.5e: Pagine html per la visualizzazione

4.3.3 Event handler

Si occupa della gestione dei dati dinamici**.** Non tutti gli event handler sono stati utilizzati nella realizzazione della applicazione BSP.

4.3.3.1 OnCreate

OnCreate è chiamato una volta solo alla creazione della pagina, ed esegue un'inizializzazione di dati o una creazione di oggetti.

4.3.3.2 OnRequest

La chiamata avviene ogni volta che si effettua una richiesta per un determinata pagina ed è utilizzato per ripristinare le strutture dei dati interne dalla richiesta. Le immagine 4.6, 4.6°, 4.6b.

```
* the handler is called whenever a request is made for a particular page
* it is used to restore the internal data structures from the request
data:
      lv_qty_to_pick  type lqua_verme,
      ls_su          type line of zwmrf_lqua_ext_tab,
      ls_component    type zht_flexwm_comp,
      lv_exit_flag    type c.

DATA:    lv_int_actual_qty      type lqua_verme,
         lv_single_su_um_qty    type lqua_verme,
         lv_roundup_value       type lqua_verme,
         lv_single              type lqua_verme.

field-symbols:     <ls_su> type zwmrf_lqua_ext.

application->pc_actual_row_idx = curr_su_idx.

read table application->gt_pc_su_grouped into application->pc_actual_su index application-
>pc_actual_row_idx.
ls_component = application->actual_comp.

select single maktx from makt into mat_text where matnr = ls_component-matnr and spras = sy-
langu.

lv_qty_to_pick = ls_component-to_stage_qty.
free application->su_selection.
loop at application->gt_pc_su into ls_su where lenum = application->pc_actual_su-lenum
  and lgtyp = application->pc_actual_su-lgtyp and lgpla = application->pc_actual_su-lgpla.

  select single * from mch1 into corresponding fields of ls_su where matnr = ls_su-matnr and charg
= ls_su-charg.
  append ls_su to application->su_selection.
```

Figura 4.6

```
endloop.
sort application->su_selection by lwedt ascending.
loop at application->su_selection assigning <ls_su>.
  clear lv_exit_flag.

  if lv_qty_to_pick > <ls_su>-pc_um_verme.
    <ls_su>-pc_prop_qty = <ls_su>-pc_um_verme.
    lv_qty_to_pick = lv_qty_to_pick - <ls_su>-pc_um_verme.
    <ls_su>-pc_prop_qty_text = trunc( <ls_su>-pc_prop_qty ).
    condense <ls_su>-pc_prop_qty_text.
  else.
    <ls_su>-pc_prop_qty = lv_qty_to_pick.
    <ls_su>-pc_prop_qty_text = trunc( <ls_su>-pc_prop_qty ).
    condense <ls_su>-pc_prop_qty_text.
    lv_exit_flag = 'X'.
  endif.
  "--> calculating roundup value
  lv_int_actual_qty = <ls_su>-pc_prop_qty.

  "--> get single su um qty in required qty
  lv_single = 1.
  DATA: check_lvsme   type MEINS.
  SELECT SINGLE LVSME FROM MLGN INTO check_lvsme WHERE MATNR = <ls_su>-matnr.

  IF check_lvsme IS INITIAL OR check_lvsme = ''.
    lv_roundup_value = lv_int_actual_qty.
  ELSE.
    if check_lvsme <> <ls_su>-pc_um.
      call function 'MD_CONVERT_MATERIAL_UNIT'
        exporting
          i_matnr              = <ls_su>-matnr
          i_in_me              = <ls_su>-pc_um      "<< UOM - in which you have quantity
          i_out_me               = check_lvsme      "<< UOM in which you want the quantity
```

Figura 4.6a.

```
      i_menge                = lv_int_actual_qty            "<< quantity
     importing
      e_menge                = lv_single_su_um_qty     "<< converted Qty
     exceptions
      error_in_application      = 1
      error                  = 2
      others                 = 3.
    lv_single_su_um_qty = ceil( lv_single_su_um_qty ).
    call function 'MD_CONVERT_MATERIAL_UNIT'
     exporting
      i_matnr                = <ls_su>-matnr
      i_in_me                = check_lvsme       "<< UOM - in which you have quantity
      i_out_me               = <ls_su>-pc_um        "<< UOM in which you want the quantity
      i_menge                = lv_single_su_um_qty            "<< quantity
     importing
      e_menge                = lv_single_su_um_qty     "<< converted Qty
     exceptions
      error_in_application      = 1
      error                  = 2
      others                 = 3.
    lv_roundup_value = lv_single_su_um_qty.
   else.
    lv_roundup_value = lv_int_actual_qty.
   endif.
  ENDIF.
  <ls_su>-roundup_qty = lv_roundup_value.
  if lv_exit_flag = 'X'.
   exit.
  endif.
 endloop.
* delete application->su_selection where pc_prop_qty = 0.
```

Figura 4.6b.

4.3.3.3 OnManupulation

Utilizzata per manipolare il flusso di dati HTTP in un secondo momento.

4.3.3.4 OnInitialization

Utilizzato principalmente per il recupero dei dati. Può anche eseguire qualsiasi programma. Ad esempio, consente di leggere i dati richiesti per la visualizzazione del BSP dal database. Come si mostra nella figura 4.3.4. Controlla se un id di un area del magazzino è stato inserito e mostra i materiali presenti

Pagina ampa_choose_material.htm Attivo

Proprietà | Layout | Handler eventi | Attributi pagine | Def. tipi

OnInitialization

```
1   * event handler for data retrieval
2
3     IF areaId <> ''.
4       application->selected_area_id = areaId.
5     ENDIF.
6
7     MOVE application->selected_area_id TO selected_area.
8     clear lt_materials.
9
10    call method zht_flexwm=>area_staging_list
11     exporting
12       i_supply_area = selected_area
13       i_user = application->USER
14     importing
15       o_staging_list  = lt_materials.
```

Figura 4.3.4.lista materiale di un area.

4.3.3.5 OnDestroy

Disponibile solo per eventi speciali.

4.3.3.6 OnInputProcessing

Questo gestore di eventi controlla ed elabora l'input dell'utente. Può anche definire la navigazione, cioè se l'utente debba essere portato alla stessa pagina o un'altra quando viene chiamato il BSP. Come mostrata nella figura.

```
* event handler for checking and processing user input and
* for defining navigation

data:    ls_field       type ihttpnvp,
         lt_fields      type tihttpnvp,
         o_return       type bapiret2,
         lv_destination  type string,
         ls_ft_ext      TYPE zwmrf_ft.

data:    lv_barcode     type string,
         lv_exit        type c,
         lv_tmp         type string,
         lv_matnr       type matnr,
         lv_charg       type char5,
         lv_vornr       type vornr.

call method request->get_form_fields
changing
  fields = lt_fields.
case event_id.
  when 'selectComponent'.

    read table lt_fields into ls_field with key name = 'barcode'.
    lv_tmp = ls_field-value.
    move lv_tmp to lv_barcode.

    read table lt_fields into ls_field with key name = 'matnr'.
    lv_tmp = ls_field-value.
    move lv_tmp to lv_matnr.
    read table lt_fields into ls_field with key name = 'charg'.
    lv_tmp = ls_field-value.
    move lv_tmp to lv_charg.

    read table lt_fields into ls_field with key name = 'vornr'.
    lv_tmp = ls_field-value.
    move lv_tmp to lv_vornr.
```

Figura 4.3.5. Codice eseguito quando l’event handler OnInputProccessing e chiamata.

```
lv_exit = ''.
IF lv_barcode IS NOT INITIAL AND lv_barcode <> ''.
  DATA: ls_mara   TYPE mara.
  "--> check barcode
  zht_flexwm=>barcode(
    exporting
      i_barcode      = lv_barcode " TYPE string
      i_user         = application->user " TYPE user
    importing
*       o_lgnum            = " TYPE lgnum
      o_mara             = ls_mara " TYPE mara
*       o_t_lqua           = " TYPE lqua_t
*       o_lagp             = " TYPE lagp
*       o_type             = " TYPE char2
*       o_gest_part        = " TYPE char1
*       o_std_batch_enable   = " TYPE char1
*       o_ean              = " TYPE char13
      o_batch            = lv_charg " TYPE char5
*       o_return           = " TYPE bapiret2_t
  ).
  IF ls_mara IS NOT INITIAL.
    lv_matnr = ls_mara-matnr.

    free application->actual_comp.
    read table lt_materials into application->actual_comp with key matnr = lv_matnr charg = lv_charg.
    IF application->actual_comp IS NOT INITIAL.
      lv_vornr = application->actual_comp-vornr.
    ELSE.
      lv_exit = 'X'.
    ENDIF.
  ELSE.
    lv_exit = 'X'.
  ENDIF.
ENDIF.
```

Figura 4.3.5a. Codice eseguito quando l’event handler OnInputProccessing e chiamata.

```
  "--> no errors
  IF lv_exit <> 'X' AND lv_matnr <> ''.
   free application->actual_comp.
   read table lt_materials into application->actual_comp with key matnr = lv_matnr charg = lv_charg vornr
= lv_vornr.
   "--> Compiling header data
   free application->header_data.
   application->header_data-order_num = application->actual_comp-vornr.
   application->header_data-matnr = application->actual_comp-matnr.
   application->header_data-charg = application->actual_comp-charg.
   application->header_data-maktx = application->actual_comp-maktx.

   lv_destination = 'ampa_select_su.htm'.
  ELSE.
   lv_destination = 'ampa_choose_material.htm?err=1'.
  ENDIF.
  navigation->goto_page( lv_destination ).

endcase.
```

Figura 4.3.5b. Codice eseguito quando l’event handler oninputproccessing e chiamata.

CONCLUSIONE

5.1 Raggiungimento degli obiettivi

Uno degli aspetti fondamentali della progettazione dell'applicazione per le aziende con dipendente abituati ad un sistema è *l'user experience*. Quindi si è cercato di perseguire l'obiettivo di offrire all'utente finale, un esperienza simile a ciò che e abituato, una fruizione semplice e immediata. Gli utilizzatori, infatti, sono persone che, non accolgono con entusiasmo modifiche a ciò che sono abituati. Per queste ragioni, è stato necessario prestare una particolare cura nell'implementazione di un'applicazione BSP con un interfaccia utente veramente facile da usare.

A causa dell'adozione di un nuovo sistema di interfacciamento dal web al gestionale SAP, l'applicazione BSP non viene più usato e sta diventando obsoleto (sono molte poche le aziende che lo adottano ancora).

5.2 Conoscenze acquisite

Lo stage mi ha permesso di acquisire conoscenze a me prima sconosciute e ho potuto approfondirne altre.

Iniziando dalle nuove tecnologie apprese:

- HTML5: lo stage mi ha permesso di apprendere questo linguaggio di markup per la strutturazione delle pagine web che è stato molto veloce da apprendere;
- CSS: descrivono come gli elementi HTML devono essere visualizzati su schermo, carta o su altri supporti.
- JAVASCRIPT: un linguaggio di programmazione orientato agli oggetti comunemente usato per creare effetti interattivi all'interno dei browser web.
- ABAP: è il linguaggio nativo di programmazione del gestionale SAP, quindi di proprietà della società SAP.

Essendo che il linguaggio ABAP consiste maggiormente nel recuperare dati dal database (query), durante il periodo dello stage ho potuto inoltre, approfondire e rinfrescare tecnologie viste durante il corso di studi come il database MySQL.

Horst Keller, Sascha Kruger, ABAP Objects - ABAP Programming in SAP NetWeaver

Rich Heilman, Thomas Jung Next generation ABAP Development

Gunther Farber, Julia Kirchner ABAP Basic

https://help.sap.com/saphelp_nw70/helpdata/EN/5a/f8b53a364e0e5fe10000000a11405a/frameset.htm

https://help.sap.com/saphelp_nw70/helpdata/EN/53/ac1e3a0088e042e10000000a11402f/content.htm

ABAP Advanced *Business Application Programming* è un linguaggio di programmazione proprietario della società SAP.

APPLICATION /APPLICAZIONE un programma o una serie di programmi in fase di esecuzione su un computer con lo scopo e il risultato di rendere possibile un servizio, una serie di servizi o strumenti utili e selezionabili su richiesta dell'utente, spesso attraverso un'elaborazione a partire da un input fornito dall'utente interagendo con esso.

BAPI Business Application Programming interfaccia è un metodo di un oggetto business SAP che può essere chiamato da un processo esterno. BAPI sono di tipo transazionale nel sistema SAP.

BUFFERING Salvataggio di dati in un buffer (spazio temporaneo) per uso futuro.

CSS Cascading Style Sheets - è un linguaggio usato per definire la formattazione di documenti HTML, XHTML e XML.

ERP è un software di gestione che integra tutti i processi di business rilevanti di un'azienda.

EVENT HANDLER Sezione di codice che corrisponde ad un'azione effettuata dall'utente.

HTML Hypertext Markup Language, un sistema standardizzato per il tagging di file di testo per ottenere effetti di font, colore, grafica e collegamenti ipertestuali sulle pagine del World Wide Web.

HTTP HYPERTEXT TRANSFER PROTOCOL usato come principale sistema per la trasmissione d'informazioni sul web.

HTTPS HyperTesxt Transfer Protocol Secure principale sistema

JAVASCRIPT è un linguaggio di scripting orientato agli oggetti comunemente usato nella creazione di siti web (programmazione Web).

MVC Model View Control Paradigma di progettazione del software che separa il software in tre parti. Model informazione salvato sul database, View visualizzazione del model e Control gestione del model.

NAMESPACE un insieme di nomi di elementi e nomi di attributi identificati univocamente da un identificatore.

PROXY un server che funge da intermediario.

RENDERING generazione di una pagina a partire da un descrizione.

RUNTIME BEHAVIOUR comportamenti durante l'esecuzione.

SAP Systems Applications and Products in Data Processing gestionale aziendale che consiste I diversi moduli che copre i diversi aspetti della gestione aziendale.

STATEFUL significa che ogni richiesta è dipendente da quelle precedenti anche se e concluso.

STATELESS significa che ogni richiesta è indipendente da quelle precedenti e si conclude al momento della chiusura della connessione.

TRANSAZIONE un codice composto da lettere, numeri o entrambi, associato a ciascuna funzione in SAP ERP.

SCRIPTING

URL Uniform Resource Locator è una sequenza di caratteri che identifica in modo univoco l'indirizzo di una risorsa in internet.

per la trasmissione d'informazioni sul web in modo sicuro.