Progettazione di interfacce web indipendenti dal dispositivo

Candidato
Izzo Giovanni, Matr. 41/1305

Relatore
Prof. Porfirio Tramontana

Panoramica su contesto ed obiettivi

Il **contesto** della tesi è legato alle nuove esigenze degli sviluppatori di applicazioni web motivate dalla gestione di utenti che :

- necessitano o scelgono di accedere alla rete internet attraverso dispositivi fissi e mobili diversi dal classico computer desktop
- necessitano di tecnologia assistiva per persone disabili

Obiettivi della tesi sono

- esplorare la possibilità di adottare una progettazione delle interfacce utente delle applicazioni web tale da utilizzare una descrizione indipendente dal dispositivo terminale
- esplorare la realizzazione di un sistema di adattamento automatico delle descrizioni per la distribuzione attraverso una varietà di reti e la fruizione attraverso diverse tipologie di dispositivi terminali.

Contesto : accessibilità e indipendenza dal dispositivo

- Dall'analisi delle linee guida Web Content Accessibility Guidelines redatte dal W3C per supportare lo sviluppo di contenuti web che siano fruibili anche da persone che utilizzano tecnologia assisitiva per disabili, si è evidenziato che molte delle problematiche dell'accessibilità possano essere ricondotte a quelle legate alla progettazione di interfacce web indipendenti dal dispositivo
- Diversi dispositivi di accesso alle applicazioni web sono caratterizzati da diverse capacità di input ed output , diversi linguaggi di marcatura e diverse tipologie di reti supportate. Queste caratteristiche possono essere raccolte in quello che è chiamato **contesto di distribuzione** di una pagina web, ed utilizzate per adattare i contenuti veicolati verso un particolare dispositivo
- In precedenti lavori di tesi sono stati descritti i diversi aspetti di cui una procedura per adattare l'interfaccia utente di una web application al contesto di distribuzione si deve interessare

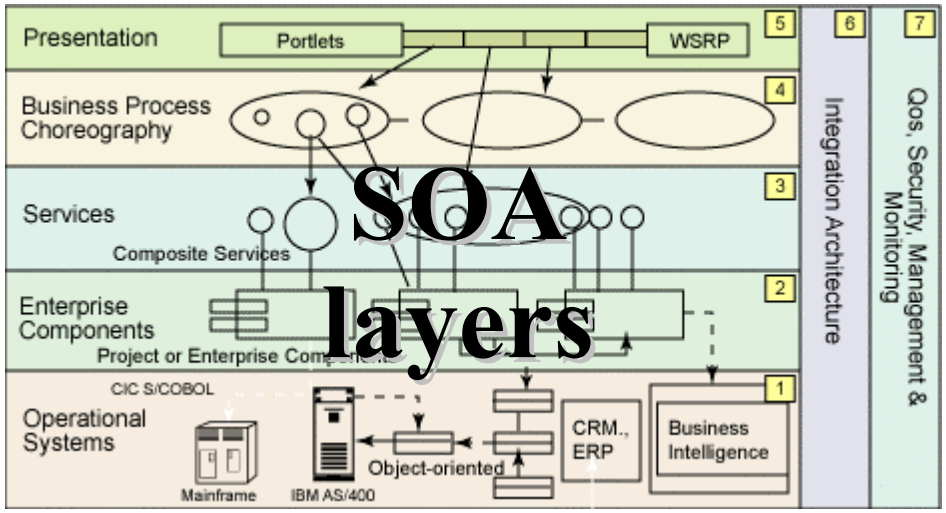
Contesto : le specifiche CC/PP e DIAL del W3C

- Attraverso le specifiche **CC/PP** è possibile formalizzare la descrizione del contesto di distribuzione, definendo lo schema di quello che è chiamato *Profilo* dell'user agent ed è costituito da una struttura dati fatta da un numero arbitrario di *Componenti* caratterizzati ciascuno da un certo insieme di coppie di *Attributi*/Valori. Ciascun componente racchiude informazioni riguardanti una caratteristica del contesto di distribuzione. Poichè CC/PP è basato su XML ed RDF è possibile automatizzare la validazione e la risoluzione delle istanze dei Profili
- Il Device Indipendent Authoring Language (**DIAL**) permette di caratterizzare in maniera indipendente dal dispositivo terminale gli elementi dell'interfaccia utente di una web application. Si avvale di opportuni sottoinsiemi di
  - *XHTML* per descrivere la struttura generale dell'interfaccia
  - *XForms* per descrivere i dialogs e gestire l'interazione con l'utente finale
  - *DISelect* per selezionare e filtrare il contenuto
  - *CSS* per descrivere la presentazione ed il layout

Contesto : le applicazioni web

- Il caso più generale di applicazione web, prevede l'eventualità che il contenuto dell'interfaccia utente sia dinamicamente generato come il risultato di una elaborazione svolta dall'applicazione stessa. Per poter caratterizzare l'interfaccia utente in questi casi, si sono combinate le caratteristiche offerte dalle specifiche DIAL con una estensione dello standard XML dell'INRIA denominata **Active Tags**, che permette di gestire
  - l'inclusione in files XML di dati dinamicamente generati
  - strutture dati complesse in maniera compatta

- L'architettura più generica ed attuale cui si può fare riferimento per la realizzazione di applicazioni distribuite complesse come possono essere le web application, è la **Service Oriented Architecture** (SOA)



- Le componenti per la gestione delle interfacce utente indipendenti dai dispositivi possono essere inserite a livello del *Presentation Layer* di una SOA

Il caso di studio

Gli obiettivi della tesi saranno perseguiti attraverso la realizzazione di un caso di studio, costituito dalla progettazione del front-end web based di una applicazione legacy (Pine: un e-mail agent) la cui migrazione verso una Service Oriented Architecture è stata già sviluppata in precedenti lavori fino alla realizzazione del Service layer, ovvero all'esposizione di alcune funzionalità come web service.

Coerentemente all'analisi preliminare, il sistema di gestione delle interfacce utente dovrà :

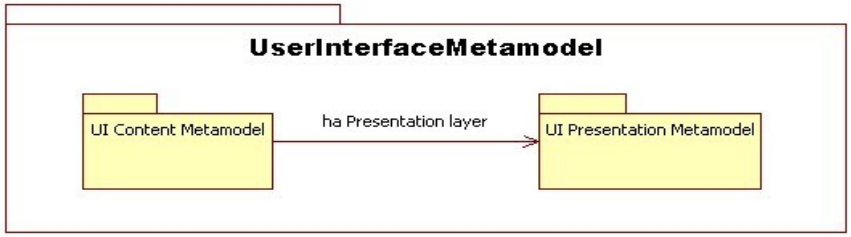
- *utilizzare gli standard del W3C (CCPP, DIAL) per implementare l'adattamento al contesto di distribuzione*
- *essere facilmente riutilizzabile all'interno di applicazioni web SOA based*
- *essere in grado di gestire l'eventualità di interfacce utente che includano dati generati dinamicamente attraverso Active Tags*

Il processo di sviluppo

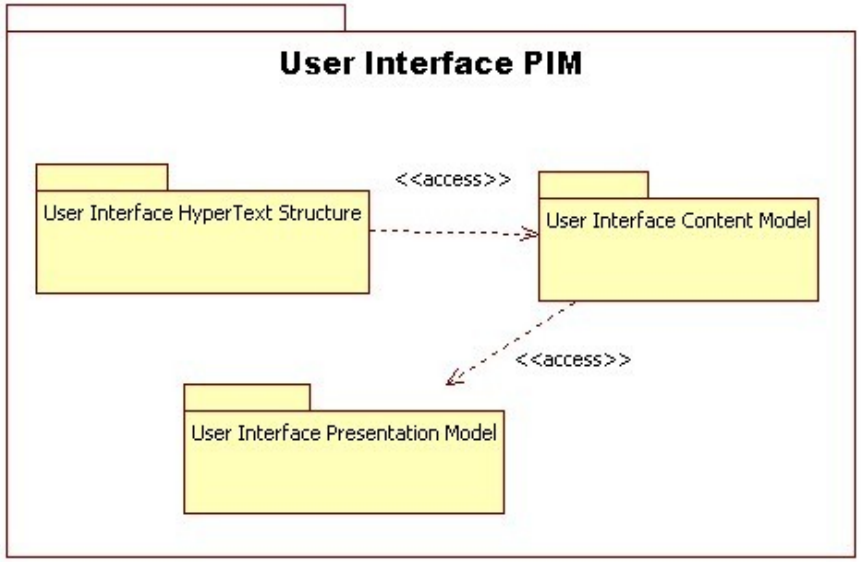
Si è ricorso ad una versione ridotta dell'***Unified Process***, adattato con una serie di elementi aggiuntivi adeguati per la progettazione del Presentation Layer di una web application SOA based

Per supportare la progettazione indipendente dal dispositivo è stata prevista la definizione di

- uno **User Interface Metamodel** che individui gli elementi significativi dell'interfaccia utente



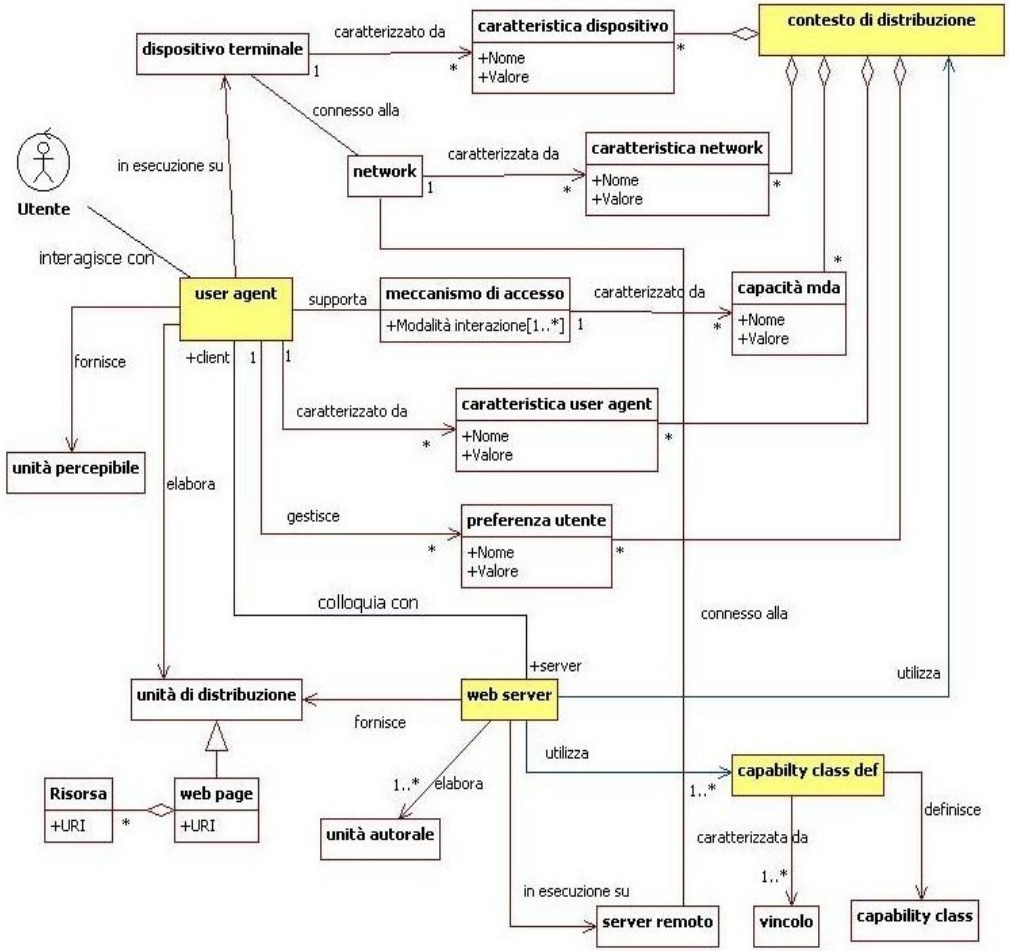
- un **Modello dell'Interfaccia Utente**, che sia un Platform Indipendent Model, e permetta di specificare
  - la struttura ipertestuale con viste e collegamenti tra le viste
  - la struttura di ciascuna vista
  - la presentazione di ciascuna vista



Caso di studio: Modello di Domino

Nella modellazione relativa al dominio dell'indipendenza dal dispositivo sono delineate:

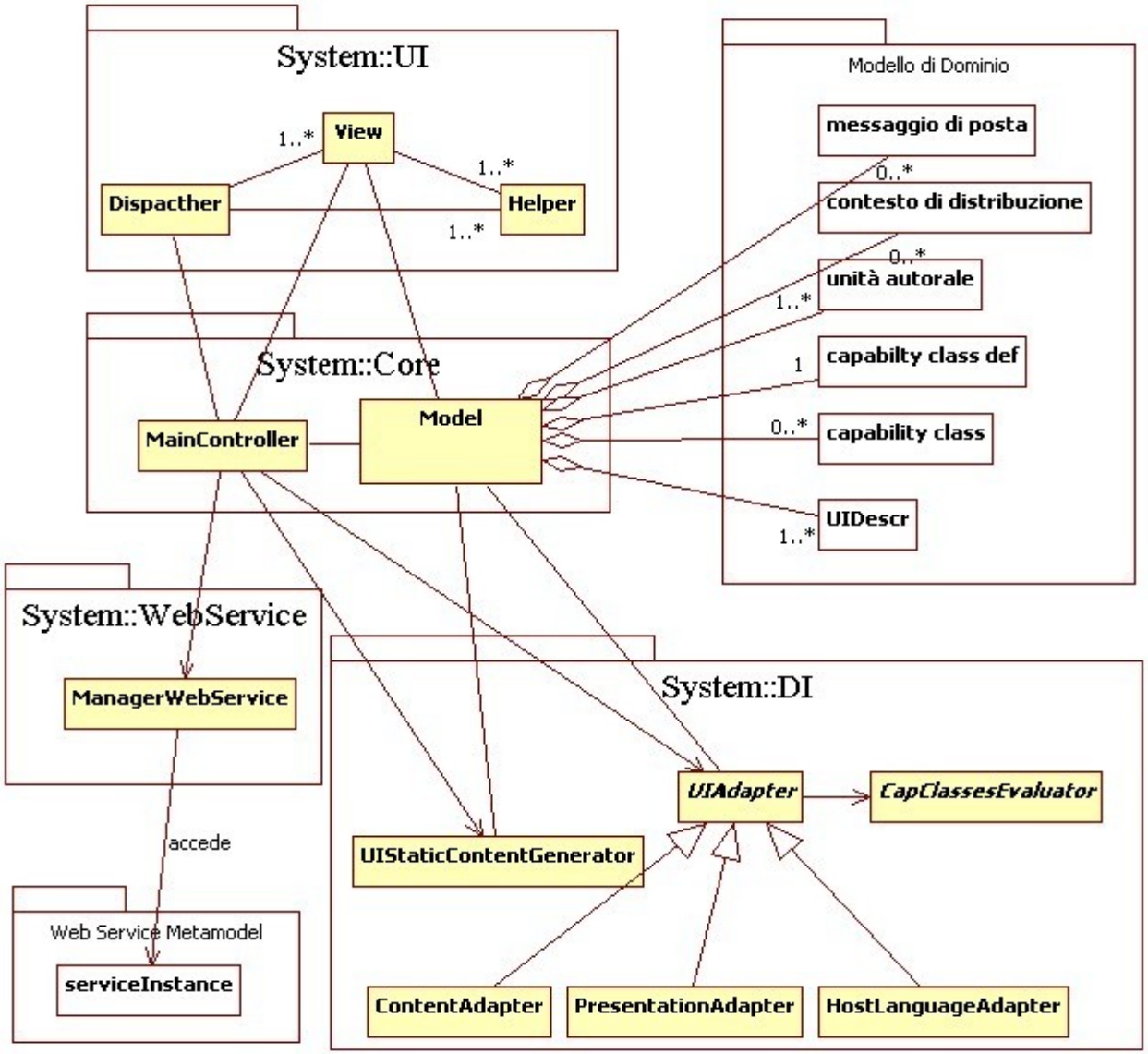
- le astrazioni significative della terminologia e del contenuto informativo del "Glossario dei termini per l'indipendenza dal dispositivo"; tra di esse, oltre al contesto di distribuzione, particolare importanza ricoprirà nel seguito il concetto di **unità autorale**
- alcune astrazioni collegate con le **classi di capacità** di un profilo CC/PP, che saranno utilizzate dal sistema di adattamento automatico delle descrizioni dell'interfaccia utente al dispositivo terminale



Caso di studio : Architettura Software

Nella **Architettura logica** sono definiti ad un alto livello di astrazione i packages in cui sono raggruppati gli elementi del sistema. In previsione dell'integrazione del sistema all'interno del presentation layer di una web application SOA based, sono stati utilizzati in fase di progetto i design patterns

- Model-View-Controller e Service-To-Worker per i packages Core &UI
- Facade per la classe ManagerWebService

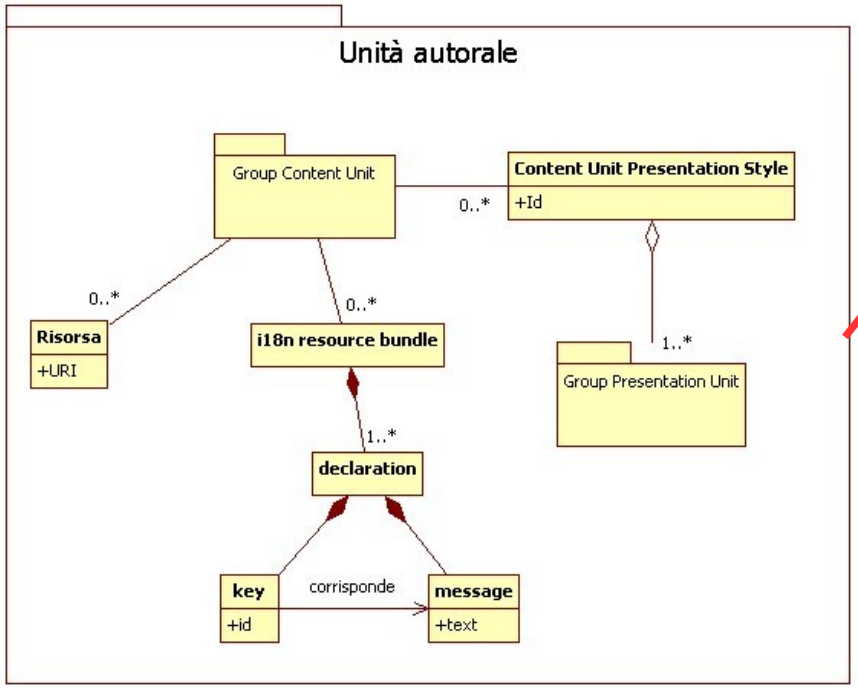


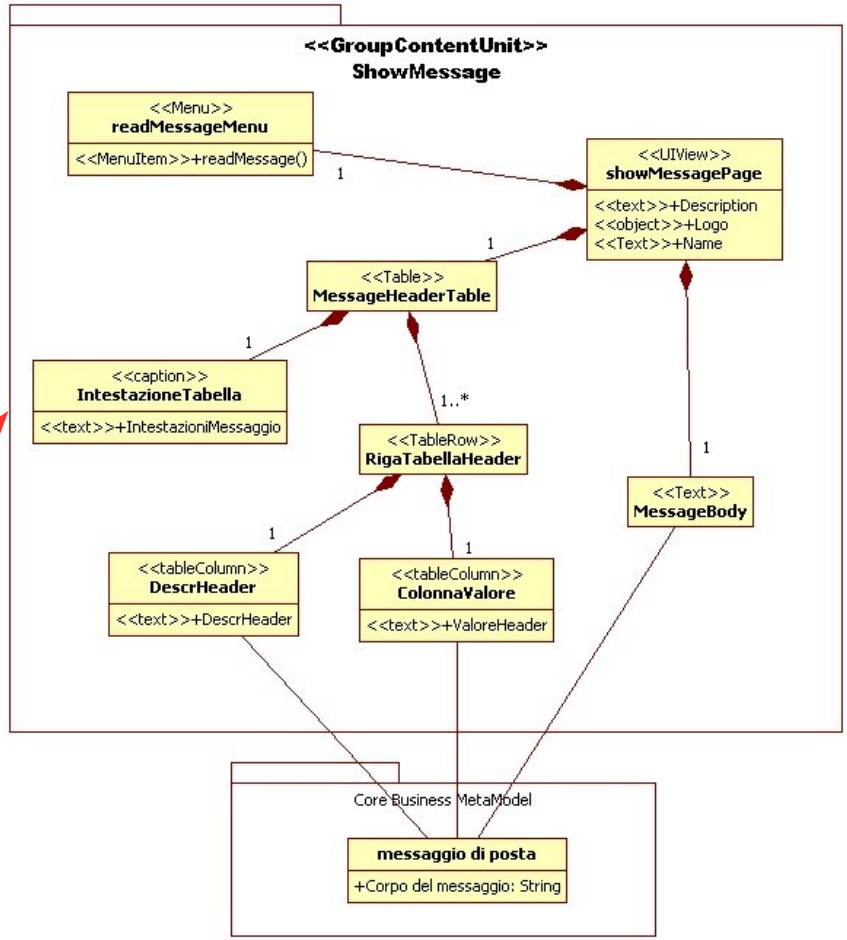
Architettura logica del sistema

Caso di studio: Modello di Progetto

Lo **User Interface Content Model** raggruppa una serie di descrizioni delle unità autorali organizzate secondo un particolare modello

Per ogni unità autorale è prevista una descrizione della **Group Content Unit** che utilizzi solo stereotipi riconducibili alle classi concettuali dello User Interface Metamodel, svincolate da una qualsiasi piattaforma implementativa

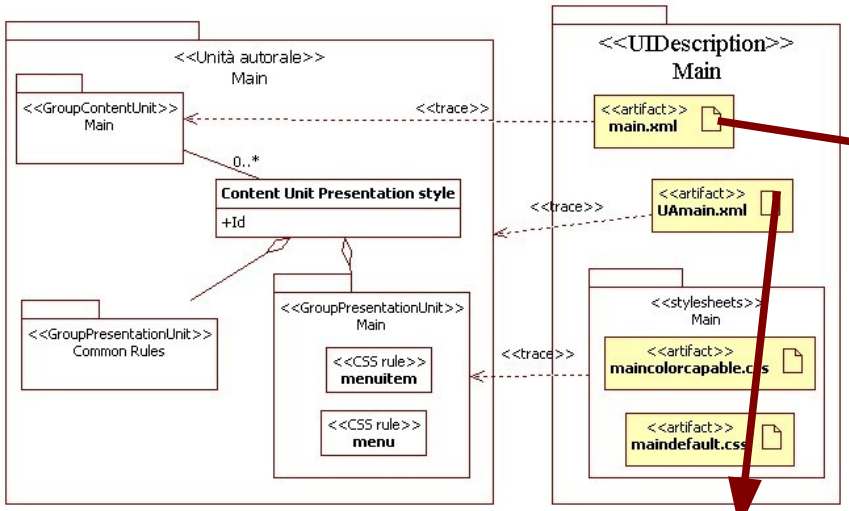




Modello della descrizione di una unità autorale

Caso di studio: Modello di Implementazione

Dalle descrizioni delle unità autorali delineate nello **User Interface Content Model** si derivano una serie di descrizioni da utilizzare all'interno del programma per la generazione delle descrizioni adattate ad un dispositivo terminale




```
<?xml version="1.0" encoding="UTF-8"?>

<UIunitaAutorale xmlns:sel='http://www.w3.org/2005/sel' id="main">
    <UIgroupContentUnit id="main" />
    <ContentUnitPresentationStyles>
        <sel:select>
            <sel:when expr="mydcn:colorcapable()">
                <ContentUnitPresentationStyle id="MainColorcapable">
                    <UIgroupPresentationUnit id="mainwithcolors" />
                    <UIgroupPresentationUnit id="commonsrulewithcolors" />
                </ContentUnitPresentationStyle>
            </sel:when>
            <sel:otherwise>
                <ContentUnitPresentationStyle id="MainDefault">
                    <UIgroupPresentationUnit id="maindefault" />
                    <UIgroupPresentationUnit id="commonsruledefault" />
                </ContentUnitPresentationStyle>
            </sel:otherwise>
        </sel:select>
    </ContentUnitPresentationStyles>
</UIunitaAutorale>
```

UAmain.xml

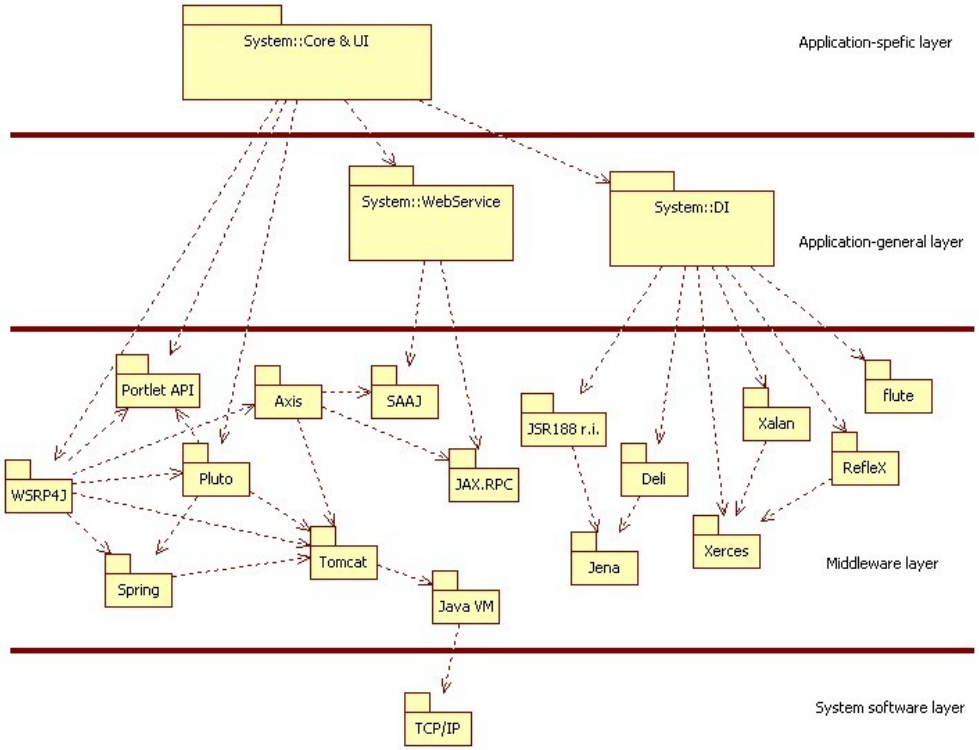
```
<?xml version="1.0" encoding="UTF-8" ?>
<UIElementGroup
  xmlns='http://xml.netbeans.org/schema/UIElement'
  xmlns:xsi='http://www.w3.org/2001/XMLSchema-instance'
  xmlns:xhtml='http://xml.netbeans.org/schema/xhtml-schema'
  xmlns:extUI='http://xml.netbeans.org/schema/ExtUISchema'
  xmlns:sel='http://www.w3.org/2005/sel'
  xmlns:dcn='http://www.w3.org/2005/dcn'>

  <xhtml:div class="portlet-section-header">
      <xhtml:p id="main.nome" xml:lang="en">
          SOAPine Web App
      </xhtml:p>
      <sel:select>
        <sel:when expr="mydcn:wmvCapable()">
            <xhtml:object type="video/wmv" id="main.logo" src="Logo.wmv" >
              main.logo
            </xhtml:object>
        </sel:when>
        <sel:when expr="mydcn:gifCapable()">
           <xhtml:object type="image/gif" id="main.logo" src="Logo.gif" >
              main.logo
           </xhtml:object>
        </sel:when>
        <sel:otherwise>
          <xhtml:object type="text/plain" class="TetxAppLogo" id="Main.Logo" src="TextAppLogo.txt"
                        style="{style:getTextAppLogoStyle()}">
                  </xhtml:objetc>
        </sel:otherwise>
      </sel:select>
      <xhtml:p id="main.descrizione" title="Descrizione" >
          main.descrizione
      </xhtml:p>
  </xhtml:div>
  <extUI:menu class="portlet-menu" id="MainMenu" description="Menu Principale">
    <extUI:menuitem class="portlet-menu-item" id="readMessage" accesskey="0" tabindex="0" />
    <extUI:menuitem class="portlet-menu-item" id="setDeliveryContext" accesskey="1" tabindex="1" />
    <extUI:menuitem class="portlet-menu-item" id="showDeliveryContext" accesskey="2" tabindex="2" />
  </extUI:menu>
</UIElementGroup>
```

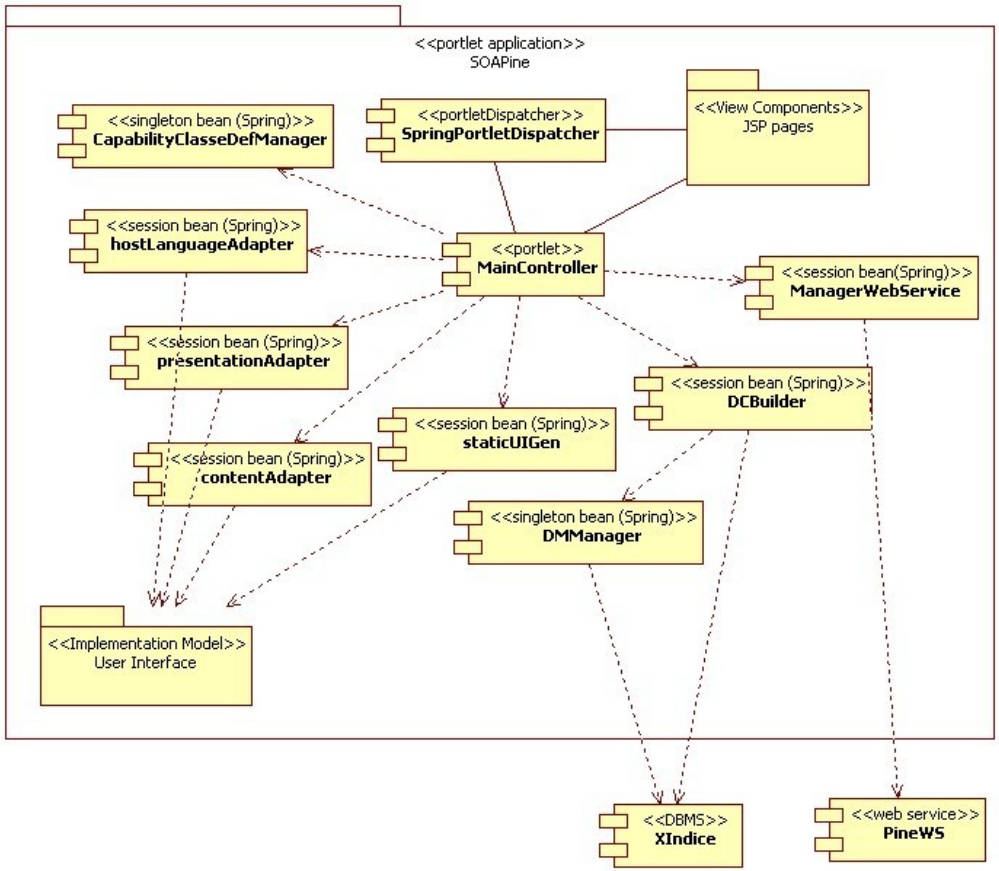
Active Document - main.xml

Dallo **User Interface Metamodel** è stato ricavato anche lo schema dati XML utilizzato per la validazione automatica delle descrizioni XML delle GroupContentUnit

Caso di studio: Modello di Implementazione

Il sistema è stato implementato come portlet application, in cui il MainController coordina la procedura di renderizzazione delle unità autorali, delegando ai componenti ausiliari la gestione dei sottosistemi del middleware layer utilizzati per realizzare l'adattamento al contesto di distribuzione



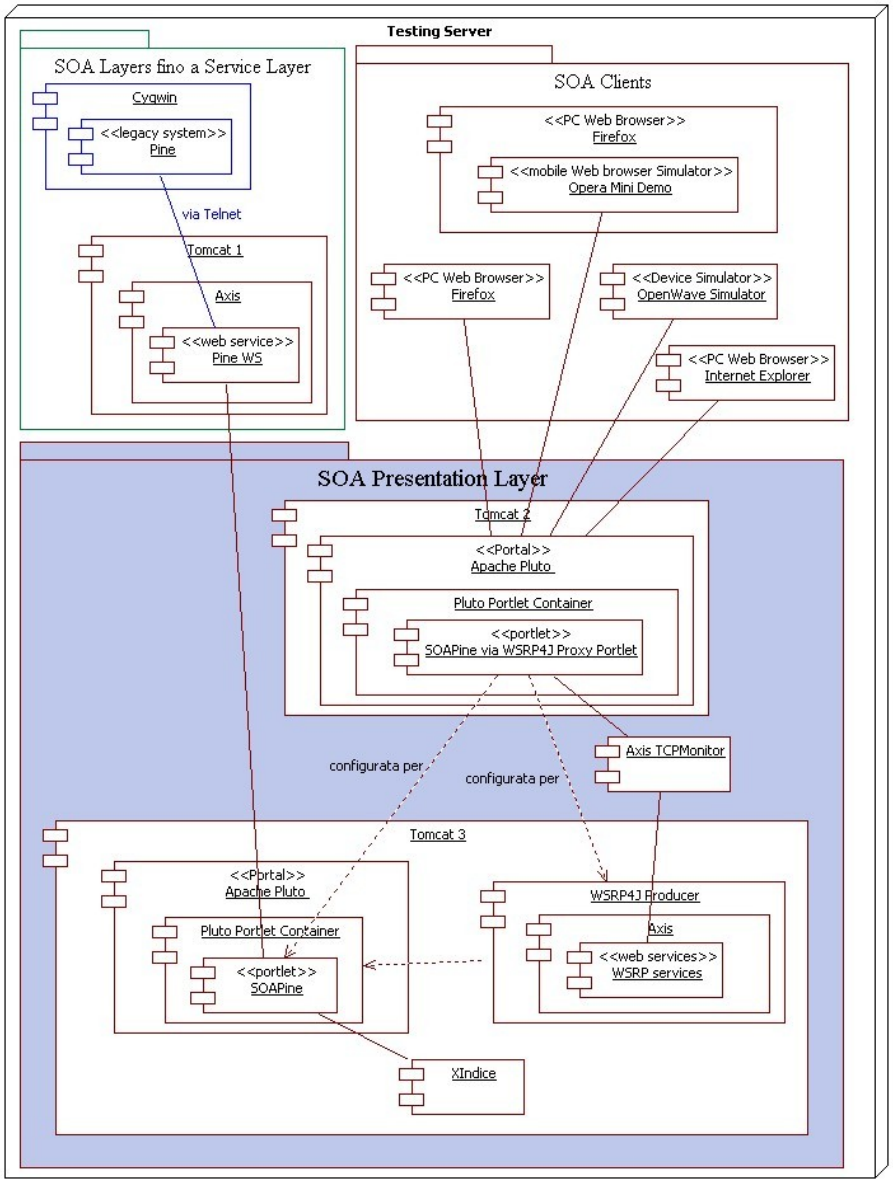


Coerentemente con l'architettura J2EE di riferimento per l'implementazione, i componenti ausiliari del MainController sono stati implementati in maggior parte come session bean stateless

Caso di studio: Modello di Deployment

Per testare il sistema è stato configurato un testing server che esegua diverse istanze indipendenti del servlet container Apache Tomcat, in cui siano messe in opera le varie componenti della portlet application e le infrastrutture di supporto alla SOA.

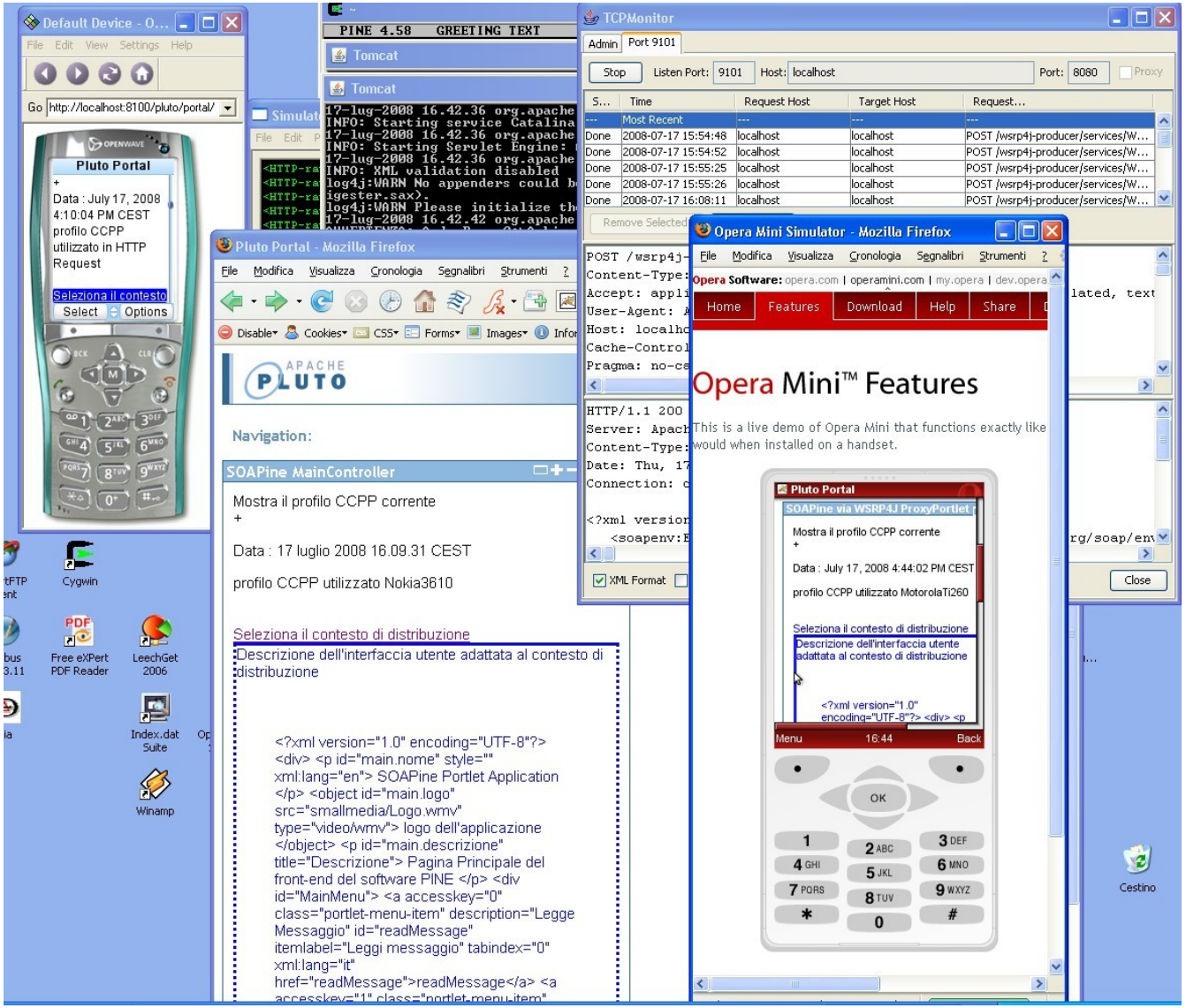
Come infrastruttura del presentation layer sono stati predisposte due installazioni del Pluto Portal in combinazione con una versione opportunamente modificata di WSRP4J



Caso di studio: Testing

Durante le prove del sistema sono state verificate

- La corretta risoluzione dei profili CC/PP sia nel caso fossero gestiti tramite il repository Xindice sia nel caso fossero indicati tra gli headers HTTP
- Il corretto evolversi della procedura di adattamento automatico delle descrizioni dell'interfaccia utente al contesto di distribuzione
- La corretta integrazione con il sistema WSRP
- La corretta gestione della multiutenza da parte della portlet application



Conclusioni

Il processo di sviluppo delineato permette di ottenere delle descrizioni dell'interfaccia utente indipendenti dal dispositivo terminale conformi agli standard individuati nell'analisi del contesto (documenti XML con elementi DIAL ed Active Tags)
Il sistema di adattamento automatico realizzato come caso di studio è in grado di elaborare queste descrizioni in funzione delle classi di capacità del contesto di distribuzioni in tempi risultati accettabili

Come sviluppi futuri

- potrebbero essere elaborati dei meccanismi di traduzione automatica dei modelli dello User Interface Content Model negli artefatti utilizzati nel Modello di Implementazione
- per permettere la corretta composizione di frammenti di markup non HTML/XHTML, si potrebbe duplicare il sistema di adattamento automatico anche all'interno dell'architettura di integrazione del Presentation Layer SOA