



Cuciniamo

Linguaggio per la gestione di un ricettario

Scenario



- ◆ Utente con conoscenze base sull'utilizzo del pc
- ◆ Vari sistemi operativi su cui il software può essere eseguito (Windows, Linux, Mac, ...)
- ◆ Disponibilità di ricette su file scritti in precedenza (anche con editor di testo esterni come Word, OpenOffice, ...)
- ◆ Necessità di un software che metta ordine in una ricetta, la cataloghi e la renda disponibile in varie visualizzazioni (per aiutare l'utente nella preparazione)

Obiettivi



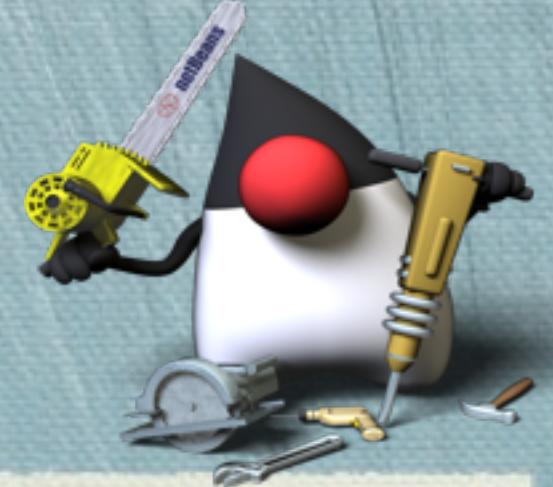
- ◆ Rappresentazione interna di una ricetta che sia
 - ◆ Semplice da interrogare
 - ◆ Facilmente estendibile
- ◆ Creazione di un linguaggio semplice e intuitivo per la scrittura di ricette che
 - ◆ Non vincoli l'utente a ricordarsi costrutti particolari
 - ◆ Utilizzi i soli caratteri stampati sulla tastiera per la grammatica
 - ◆ Ponga vincoli leggeri sull'ordine con cui possano essere passate le informazioni e sia case insensitive
- ◆ Creazione di una buona interfaccia grafica di supporto
- ◆ Consentire il supporto futuro ad un database remoto per il salvataggio e/o reperimento di ricette online e permettere l'esportazione in XML (bozza di validazione sia in DTD che XSD disponibile)

Analisi del problema



- ◆ Si dovrà quindi realizzare un compilatore per tale linguaggio che prenda in input una stringa di caratteri
 - ◆ Inserita direttamente nella finestra del programma
 - ◆ Caricata da un file di testo
- ◆ Esso dovrà effettuare l'analisi sintattica e di seguito quella semantica
- ◆ Nel caso tutte le operazioni vadano a buon fine verrà prodotta una rappresentazione adeguata
- ◆ In caso di errore, questo verrà segnalato all'utente nel modo più chiaro e dettagliato possibile
- ◆ Il tutto tenendo conto degli obiettivi prefissati

Strumenti utilizzati



- ◆ Java 6 come valido e supportato linguaggio multipiattaforma
- ◆ JavaCC 5.0 per la generazione del parser del linguaggio
- ◆ J.T.B. 1.3.2 come precompilatore per la generazione del Visitor associato ad una ricetta
- ◆ Eclipse 3.6 (Helios) corredato di Ant e plugin JavaCC per lo sviluppo delle funzionalità fondamentali
- ◆ NetBeans 7 per la creazione dell'interfaccia grafica e rifiniture finali
- ◆ Libreria JGraph per graficare una ricetta
- ◆ Libreria MySql-Connector per la connessione a database remoti

Cos'è una ricetta? (1 / 2)



Il primo passo indispensabile nello svolgimento del progetto è la modellazione di una ricetta dalla prospettiva del programmatore:

- ◆ Una ricetta è un “qualcosa” che ha un nome
- ◆ Può avere alcune informazioni aggiuntive predefinite (come la categoria d'appartenenza, il tempo di svolgimento, la difficoltà, ...)
- ◆ Può necessitare di altre (sotto)ricette per il suo corretto completamento
- ◆ E' sicuramente composta da un insieme di ingredienti e da una descrizione di come debbano essere combinati per ottenere il risultato desiderato (con eventuali immagini di supporto)

Cos'è una ricetta? (2 / 2)



Nel dettaglio si è scelto di rappresentare:

- ◆ Il nome di una ricetta
- ◆ Il tempo di preparazione
- ◆ La categoria d'appartenenza
- ◆ La difficoltà d'esecuzione
- ◆ L'autore della stessa
- ◆ Gli strumenti usati
- ◆ La festività d'appartenenza
- ◆ Un veloce promemoria
- ◆ Ingredienti e quantità
- ◆ Il procedimento necessario
- ◆ I link alle immagini d'aiuto

E' tutto?

- ◆ Assolutamente no! E' solo quello che si è deciso di modellare!
- ◆ Volendo la lista può ampliarsi... (descrizione introduttiva, numero di dosi, tempi di cottura, costo, calorie, ...)

Un primo esempio



Torta alla panna

Salva: tortapanna

Autore: ptommasi

Festività: San Valentino

Categoria: Torte (Dolci)

Difficoltà: 7

Tempo: 1 ora

Prepara: crema

Salva: crema

Tempo: 30 minuti

Strumenti: fruste

Promemoria: a fiamma bassa

Ingredienti: 6 tuorli, 220g zucchero, 120g farina, 1l latte, 2 scorze di limone

Preparazione: frulla prima i tuorli delle uova con zucchero e vanillina, poi metti la farina, poi il latte poco alla volta. Puoi usare le fruste elettriche ma a velocità minima._

Prepara: pan di spagna

Salva: pandispagna

Autore: giallozafferano

Tempo: 1 ora

Promemoria: forno a 180 gradi

Strumenti: teglia quadrata, fruste elettriche, setaccio

Ingredienti: 75g di farina, 75g di fecola di patate, 5 uova, 1 bustina di vanillina, 1 pizzico di sale, 150g zucchero

Preparazione: Per la preparazione del pan di spagna, dividete gli albumi dai tuorli [immagine1.jpg; albumi divisi] mettendoli in due ciotole grandi separate [immagine2.jpg]; iniziate a sbattere con l'aiuto delle fruste elettriche i tuorli con metà zucchero [immagine3]. Sbattete ora anche gli albumi [immagine5.jpg] e aggiungete il restante zucchero [immagine6.jpg] e proseguite a montare ancora per qualche minuto .Unite gli albumi montati ai tuorli montati e a questo punto aggiungete la farina e la fecola di patate setacciate insieme [immagine7.jpg]. Mescolate il tutto con un cucchiaio di legno fino ad ottenere un composto omogeneo, facendo attenzione a non smontarlo.

Imburrate [immagine8.jpg] e infarinate [immagine9.jpg] per bene una teglia rotonda a cerniera del diametro di 24 cm e versate l'impasto [immagine10.jpg] al centro dello stampo livellandolo per bene [immagine11.jpg].

Fate preriscaldare il forno a 180° e infornate il vostro pan di spagna per almeno 35-40 minuti senza mai aprire il forno nella prima mezz'ora di cottura.

Estraete lo stampo dal forno [immagine12.jpg; pan di spagna pronto] e fate raffreddare il pan di spagna nello stampo prima di aprirlo._

Ingredienti: panna

Preparazione: Taglia a metà il pan di spagna, riempilo con la crema, guarnisci con la panna.

Grammatica (1/5)



- ◆ Grammatica strutturata in quattro sezioni:
 1. Nome della ricetta
 2. Informazioni opzionali (inserite in ordine qualsiasi, maggiore è il pool di informazioni inserite, maggiori saranno le stime calcolabili dal programma)
 3. Eventuali richiami ad altre ricette indispensabili per il completamento della principale
 4. Ingredienti e preparazione
- ◆ Possibilità di inserire commenti che saranno ignorati dal parser

Grammatica (2/5)



◆ Scopo della grammatica:

1: Nome della ricetta 2: Informazioni opzionali 3: Sottoricette

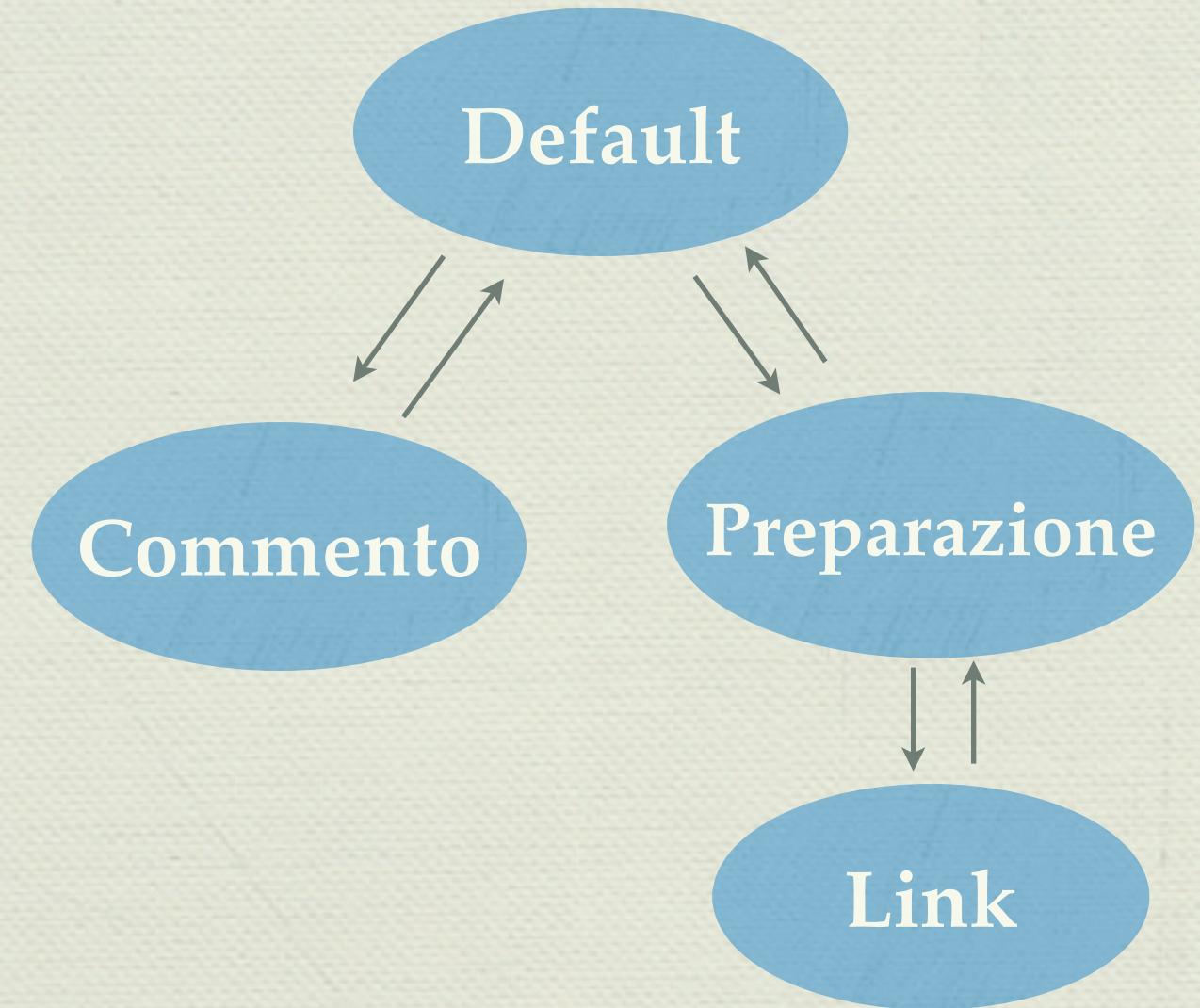
Ricetta ::= **NomeEsteso <SEPARATORE>** **Opzioni <SEPARATORE>)*** **(Sottoricetta <SEPARATORE>)***

<_INGREDIENTI> **ListaIngredienti <SEPARATORE>** <_PREPARAZIONE> **Preparazione(<FINENOTA> | <EOF>) (<FINE> | <EOF>)**

4: Ingredienti e preparazione

◆ Quattro possibili stati lessicali:

- ◆ **Default:** non tutti i caratteri sono accettati (prime tre sezioni)
- ◆ **Commento:** ignora una sequenza inline di caratteri (è possibile entrarci solo dalle prime tre sezioni)
- ◆ **Preparazione:** accetta ogni tipo di carattere (quarta e ultima sezione)
- ◆ **Inserimento link:** durante la preparazione vengono inserite immagini



Grammatica (3 / 5)



I token (con i conseguenti cambi di stato lessicale) utilizzati risultano essere:

```

TOKEN:
{
    < SEPARATORE: (["\n","\r",",",";","."] )+ >
    | < FINE: "_" >
    | < NUM: ("1"- "9")(["0"- "9"])* >
    | < STRING: ("a"- "z","A"- "Z","à","è","í","ò","ù")+ >
    | < QUANTITA: ("1"- "9")(["0"- "9"])*(["a"- "z","A"- "Z"])* >

    // Lista di token opzionali:
    | < _SALVA: "SALVA:" >
    | < _AUTORE: "AUTORE:" >
    | < _CATEGORIA: "CATEGORIA:" >
    | < _DIFFICOLTA: ("DIFFICOLTA!"|"DIFFICOLTÀ!"|"DIFFICOLTA:") >
    | < _TEMPO: "TEMPO:" >
    | < _PRIVATA: "PRIVATA:" >
    | < _STRUMENTI: "STRUMENTI:" | "USA:" >
    | < _FESTIVITA: ("FESTIVITA!"|"FESTIVITÀ!"|"FESTIVITA:") >
    | < _PROMEMORIA: "PROMEMORIA:" >

    // Token che richiamano lo scopo:
    | < _PREPARA: "PREPARA:" >
    | < _APRI: "APRI:" >

    // Token obbligatori:
    | < _INGREDIENTI: "INGREDIENTI:" >
    | < _PREPARAZIONE: "PROCEDIMENTO:" | "PREPARAZIONE:" >:
        StatoPreparazione
}

```

```

SKIP:
{ "" | "\t" }

< StatoPreparazione >
TOKEN:
{
    < NOTA: (~["_","[","]"])+ >
    | < FINENOTA: "_" > { input_stream.backup(1); } : DEFAULT
    | < _OPENLINK: "[" > : StatoInserimentoLink
}

< StatoInserimentoLink >
TOKEN:
{
    < ADVSTRING: ("0"- "9","a"- "z","A"- "Z",":",";","\","/", ".", "-",".","(",")","à","è","í","ò","ù")+ >
    | < _STARTDESC: ";" >
    | < _CLOSELINK: "]" > : StatoPreparazione
}

// Stato lessicale per i commenti in-line
SKIP: { "(" : StatoCommento }
< StatoCommento > SKIP: { ")" : DEFAULT }
< StatoCommento > SKIP: { < ~[] > }

```

Grammatica (4 / 5)



- ## 1. Il nome della ricetta (prima sezione)

NomeEsteso ::= (<STRING>)+

- ## 2. Le informazioni aggiuntive (seconda sezione):

Opzioni ::= (*<_SALVA> NomeSalvataggio* **|** *<_AUTORE> Autore* **|** *<_CATEGORIA> TipoCategoria* **|** *<_DIFFICOLTA> ValoreDifficoltà* **|** *<_TEMPO> Tempo* **|** *<_PRIVATA> Privata* **|** *<_STRUMENTI> ListaStrumenti* **|** *<_FESTIVITA> Festività* **|** *<_PROMEMORIA> Promemoria* **)**

◆ Che singolarmente sono definite come segue

NomeSalvataggio ::= <STRING> → Nome con cui si vuole salvare la ricetta sul disco (es: *tortaallapanna*, ...)

Autore ::= <STRING> → Autore della ricetta (es: *giallozafferano*, *utentex*, ...)

TipoCategoria ::= CategoriaInf ("(" CategoriaSup ")")? → Categoria di appartenenza della ricetta (es: *Dolci*, *Secondi (Carne)*, *Primi*, *Risotti*, ...)

CategoríaSup ::= (<STRING>)+

ValoreDifficoltà ::= <NUM> → Difficoltà di esecuzione della ricetta (es: 7)

Tempo ::= TempoSup ((<STRING>)? TempoInf)? —————> Tempo di preparazione (es: *1 ora e 40 minuti*, *30 minuti*, *2 ore*)

TempoSup ::= Durata UnitaTempo

TempoInf ::= Durata UnitaTempo

Durata ::= <NUM>

UnitaTempo ::= <STRING>

Privata ::= <STRING> → Se una volta inserita nel database sarà a disposizione di tutti o del solo autore

ListaStrumenti ::= Strumento ("," Strumento)* → Lista di strumenti necessari per la preparazione separati da virgola (es: *fruste*, *spatole*, ..., *attrezzi*)

Strumento ::= (<STRING>)+ elettriche, tortiera con cerniera, ...

Festivita ::= (<STRING>)+

Promemoria ::= (<STRING> | <NUM>)+ → Piccolo appunto immediato per la preparazione della ricetta

Grammatica (5 / 5)



3. L'inserimento di sottoricette (terza sezione) è possibile in due modi, o richiamando lo scopo oppure richiamando un file esterno (o uno stream da database, in una futura implementazione) che verrà immediatamente analizzato e inserito:

Sottoricetta ::= (*<_PREPARA> Ricetta* | *<_APRI> Apri*)

Apri ::= **NomeSottoricetta** (" (" **AutoreSottoricetta** ")")?

NomeSottoricetta ::= <STRING>

AutoreSottoricetta ::= <STRING>

4. Ingredienti e preparazione (quarta e ultima sezione):

ListaIngredienti ::= **Ingrediente** ("," **Ingrediente**)*

Ingrediente ::= **QuantitaIngrediente** **TipoIngrediente** | **TipoIngrediente** (**QuantitaIngrediente**)?

TipoIngrediente ::= (<STRING>)+

QuantitaIngrediente ::= <QUANTITA> | <NUM>

Preparazione ::= (**Nota** | *<_OPENLINK> Link* (*<_STARTDESC> DescrizioneLink*)? *<_CLOSELINK>*)+

Nota ::= <NOTA>

Link ::= <ADVSTRING>

DescrizioneLink ::= <ADVSTRING>

Linguaggio



- ◆ La **grammatica**, secondo la classificazione di Chomsky, è di **tipo 2** (context-free), cioè:
$$A \rightarrow \alpha$$

con $\alpha \in (VT \cup VN)^*$, $A \in VN$
- ◆ Il **linguaggio** generato è anch'esso di **tipo 2** in quanto è presente self-embedding, ovvero presenza di simboli non terminali autoinclusivi del tipo:
$$A \rightarrow \beta A \delta$$

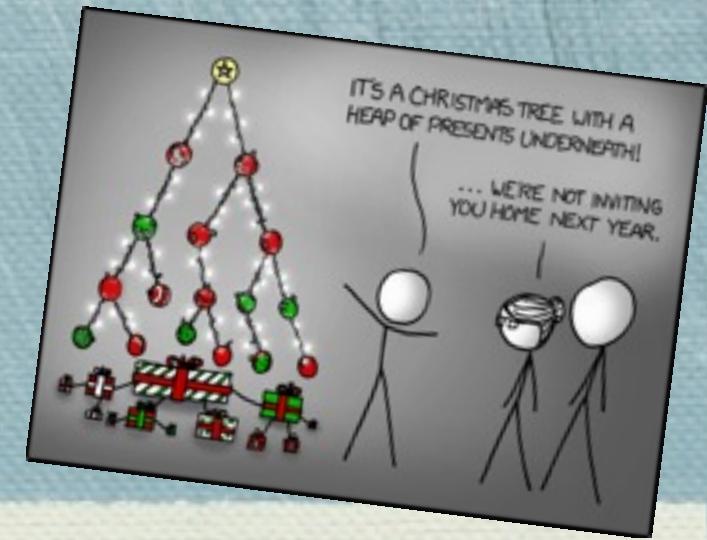
dove $A \in VN$, $\beta, \delta \in V^+$
- ◆ **LL(1)** perché nessuna produzione genera stringa vuota e gli insiemi degli starter symbols corrispondenti alla parte destra delle produzioni alternative di uno stesso metasimbolo sono tra loro disgiunti
- ◆ Si ha un **PDA deterministico** (ASF con stack) con analisi top-down

Controlli semantici



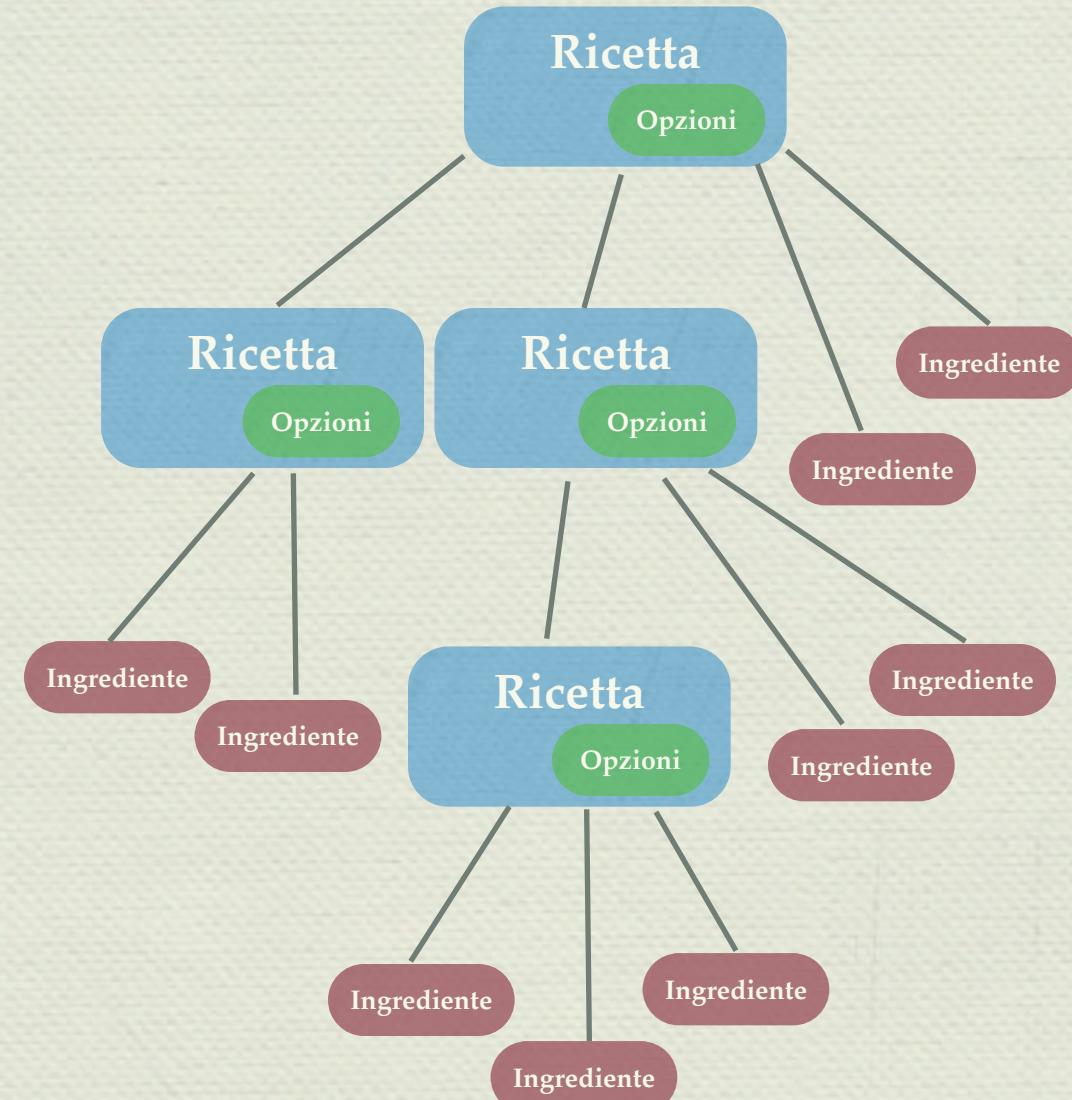
- ◆ La grammatica è molto restrittiva sui caratteri utilizzabili
⇒ controlli semantici sui tipi resi inutili
- ◆ Sgravando l'utente dal ricordare un ordine prestabilito con cui immettere le informazioni aggiuntive non è più possibile (attraverso la sola grammatica) impedire che un'informazione venga inserita più volte
⇒ semanticamente si controlla che un'opzione venga inserita al più una volta (generando un warning in caso contrario)
- ◆ Se un'opzione sfora dai valori consentiti (e.g. "difficoltà: 11") viene generato un warning e l'opzione ignorata

L'albero astratto



Ogni ricetta viene salvata sotto forma di albero, le regole con cui viene generato sono le seguenti:

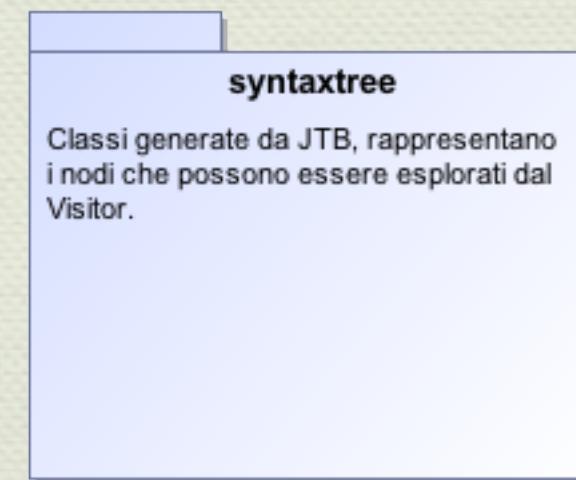
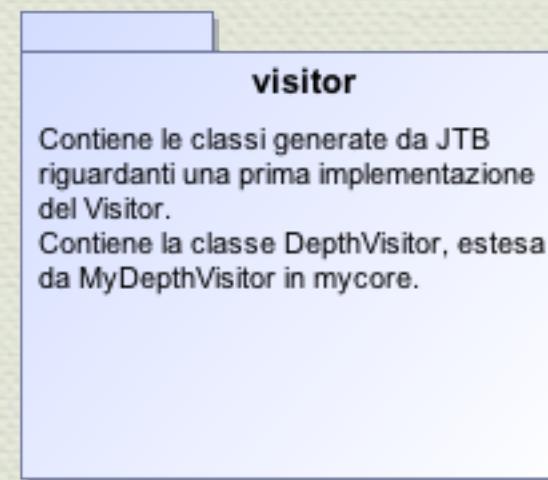
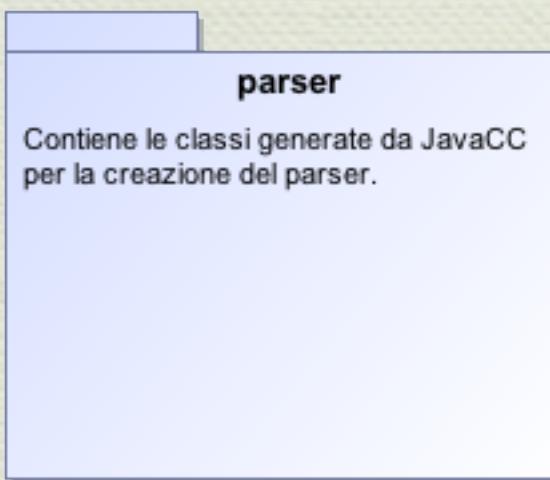
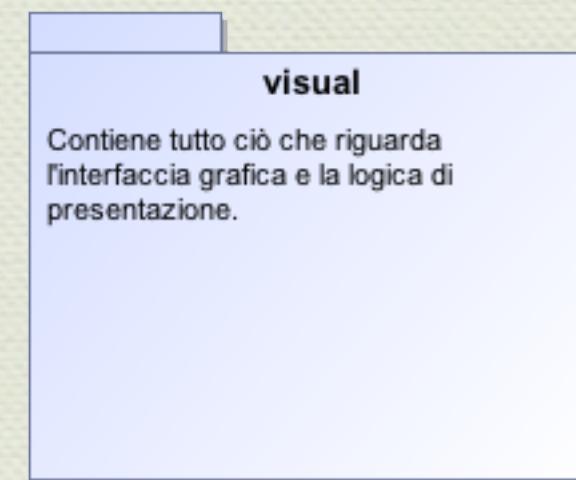
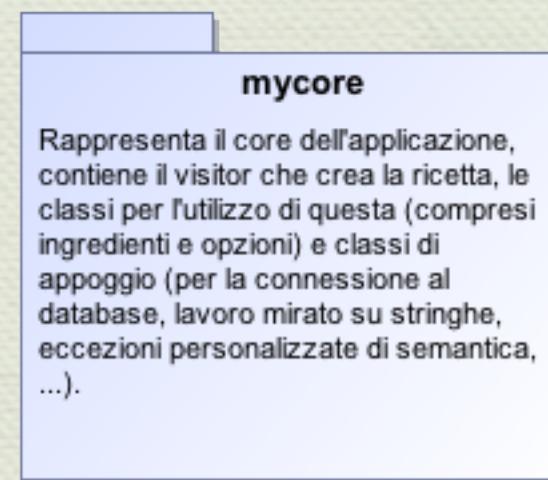
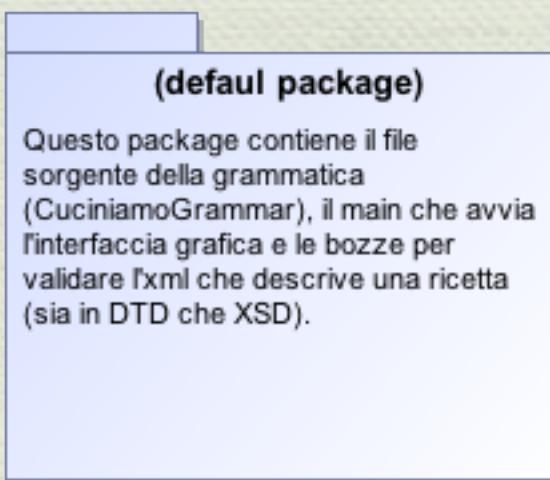
- ◆ Ogni ricetta (e quindi sottoricetta) è un nodo dell'albero
- ◆ Ogni informazione aggiuntiva inserita è un'opzione in più fornita al nodo
- ◆ Ogni ingrediente è una foglia della ricetta a cui appartiene



Implementazione



- ◆ Per un facile mantenimento il progetto è diviso in sei package (di cui tre autogenerati):



Dettagli implementativi



- ◆ Perno dell'architettura risulta essere la classe **MyRicetta**, creata inizialmente da **MyDepthVisitor** (estensione di **DepthVisitor** creata da JTB)
- ◆ Le peculiarità sono:
 - ◆ Derivazione da **TreeModel** per presentarsi come albero
 - ◆ Derivazione da **MyGraphModel** per essere disegnata sotto forma di semplice grafo (ogni nodo fornisce informazioni su profondità massima sottostante e ampiezza, ovvero numero pesato dei figli)
 - ◆ Caching dei risultati che coinvolgono chiamate ricorsive
- ◆ Inoltre al suo interno la classe **MyRicetta** contiene:
 - ◆ Un'istanza di **MyOpzioni** per tenere traccia delle informazioni aggiuntive inserite (che la sgrava dal compito di lavoro su queste)
 - ◆ Un vettore (**ArrayList** dimensionato in maniera esatta) di **MyIngrediente**

Agevolazioni software



Il software permette di:

- ◆ Calcolare la totalità degli ingredienti necessari con relative quantità
- ◆ Disegnare un grafico che accompagni l'utente nella reale preparazione

Inoltre le informazioni aggiuntive, quando presenti, permettono di:

- ◆ Salvare automaticamente la ricetta sul disco (e, in un'implementazione futura, su database)
- ◆ Catalogare una ricetta in base alla categoria (e eventuale sottocategoria) e/o festività d'appartenenza
- ◆ Fare una stima del tempo necessario per la preparazione
- ◆ Trovare la difficoltà d'esecuzione effettiva
- ◆ Memorizzare un vasto numero di informazioni utili

Interfaccia grafica (1 / 2)



La GUI, presentandosi come un ambiente di sviluppo in miniatura, punta ad essere intuitiva e ricca di funzionalità:

- ▶ Menu con tutte le opzioni possibili (con reference ed inserimento guidato dei token)
- ▶ Toolbar con le sole opzioni di più frequente utilizzo
- ▶ Editor con evidenziazione delle parole chiave e conteggio delle linee
- ▶ Area di output per guidare l'utente (con errori in rosso, warning in giallo, successi in verde e informazioni in blu)
- ▶ Albero post-compilazione con la ricetta in analisi
- ▶ Shortcut da tastiera per velocizzare le operazioni più comuni
- ▶ Stampa di un grafico che accompagni l'utente nello svolgimento pratico della ricetta
- ▶ Look & Feel a tema (con ottimizzazione per utenti Mac OS)

Interfaccia grafica (2 / 2)



The screenshot shows the Cuciniamo application window. The menu bar includes 'Cuciniamo', 'File', 'Ricetta', 'Visualizza', 'Inserisci', and '?'. The toolbar below has icons for 'Apri' (Open), 'Salva' (Save), 'Compila' (Compile), 'Guida' (Help), 'Info' (Information), and 'Esci' (Exit). The main area contains a list of recipe steps and a detailed tree view of ingredients for a specific recipe.

1 torta alla panna
2
3 Salva: tortapanna
4 Autore: ptommasi
5 Festività: San Valentino
6 Categoria: Torte (Dolci)
7 Difficoltà: 7
8 Privata: si
9 Strumenti: marisa
10 Tempo: 1 ora
11
12 Prepara: crema
13 Salva: crema
14 Tempo: 30 minuti
15 Strumenti: fruste
16 Promemoria: a fiamma bassa

torta alla panna
crema
pan di spagna
farina (75g)
fecola di patate (75g)
uova (5)
vanillina (1bustina)
sale (1pizzico)
zucchero (150g)
panna

pan di spagna: Per la preparazione del pan di Spagna, dividete gli albumi dai tuorli (rif: immagine 1) mettendoli in due ciotole grandi separate (rif: immagine 2); iniziate a sbattere con l'aiuto delle fruste elettriche, o della planetaria, i tuorli con metà dello zucchero (rif: immagine 3) fino ad ottenere un composto spumoso, gonfio e di colore giallo chiaro (rif: immagine 4). Sbattete ora, dopo aver pulito molto bene le vostre fruste, anche gli albumi ((rif: immagine 5)) e dopo circa 5 minuti quando saranno già abbastanza gonfi e bianchi, aggiungete il restante zucchero (rif: immagine 6) e proseguite a montare ancora per qualche minuto (il trucco per sapere se i bianchi sono montati a neve davvero ferma sta nell'inclinare leggermente la ciotola usata, se il composto scivola ancora

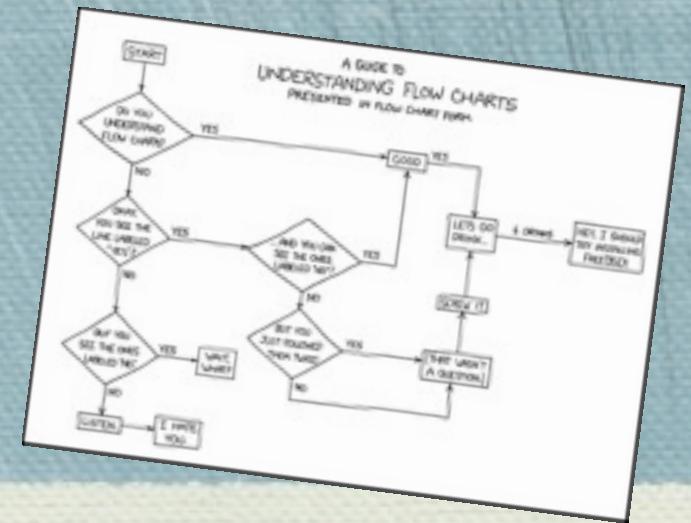
- ◆ Menu
- ◆ Toolbar
- ◆ Editor
- ◆ Albero creato
- ◆ Output

Grafico (1 / 2)

- ▶ torta alla panna
- ▼ crema
 - tuorli (6)
 - zucchero (220g)
 - farina (120g)
 - latte (1l)
 - scorze di limone (2)
- ▼ pan di spagna
 - farina (75g)
 - fecola di patate (75g)
 - uova (5)
 - vanillina (1bustina)
 - sale (1pizzico)
 - zucchero (150g)
 - panna



Grafico (2/2)

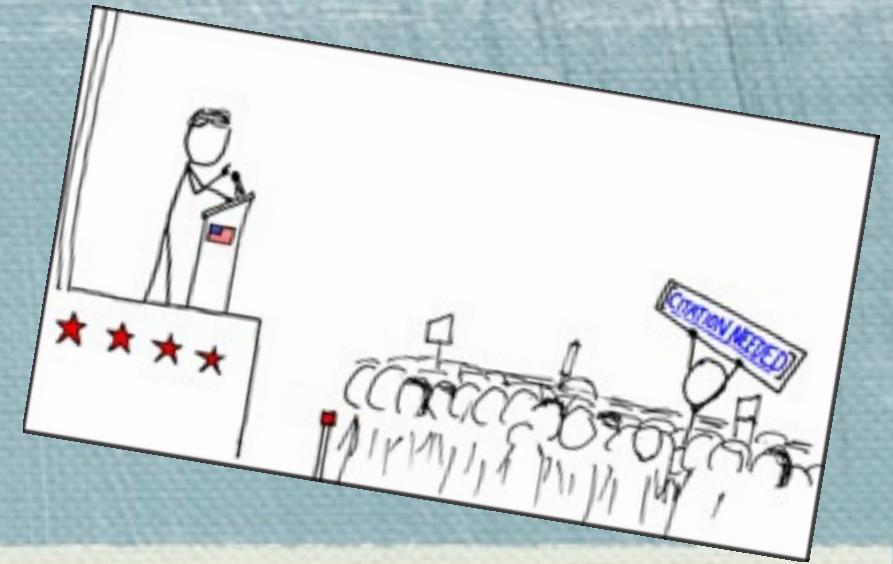


- ▶ Pasta zucca e salsiccia
 - ▼ salsicce tirate con il vino
 - salsicce (4)
 - vino bianco
 - olio qb
 - ▼ soffritto di cipolle e zucca
 - cipolla (50g)
 - zucca a cubetti piccoli (1piatto)
 - rosmarino qb
 - sale qb
 - ▼ brodo vegetale
 - dadi star vegetale (2)
 - acqua (1l)
 - panna fresca (250ml)
 - pasta fresca (400g)

busta fresca (400g)
barattolo fresco (520ml)
scatola (1kg)



Sviluppi futuri



- ◆ Appoggio ad un sito di cucina a cui fare riferimento per reperire ricette già pronte oppure soli impasti base
- ◆ Salvataggio/esportazione in XML (in vista di un'eventuale interazione con un sito di cucina, bozza di validazione sia DTD che XSD disponibile)
- ◆ Sfruttamento completo di un database (non per soli task di controllo ma anche per effettuare salvataggio, caricamento, autenticazione, ...)
- ◆ Navigazione interattiva del grafico (azioni legate al click dell'utente sul nodo, modifica delle informazioni riguardanti una ricetta/ingrediente, ...), anche in altra tecnologia maggiormente web-oriented (Flash, HTML5, ...)
- ◆ Porting del parser in PHP (progetto Marpa)

Grazie per l'attenzione

