

SALINAC: un sistema di supporto all'apprendimento di Linguaggio Naturale Controllato¹

Roberto BASILI, Maria Teresa PAZIENZA, Marco PENNACCHIOTTI

Dipartimento di Informatica, Sistemi e Produzione

Università di Roma Tor Vergata

{basili, pazienza, pennacchiotti}@info.uniroma2.it

La definizione di informazioni tecniche, quali ad esempio le specifiche del software o di dispositivi elettrici, richiede un livello molto alto di accuratezza. L'adozione (sin dalle prime fasi di progettazione) di linguaggi formali impedirebbe il coinvolgimento di personale poco specializzato elevando i costi di produzione. L'uso del linguaggio naturale risulta adatto al compito e utilizzabile dai più anche se debba essere condizionato a rigorose regole d'uso per limitarne l'ambiguità: ne derivano i cosiddetti linguaggi naturali controllati.

In questo lavoro si presenta una piattaforma software per il supporto all'apprendimento di "linguaggi naturali controllati" mediante un controllo intelligente delle attività di scrittura. Il metodo segue un approccio basato su tecniche robuste di elaborazione del linguaggio naturale e su una rappresentazione esplicita del dominio applicativo (ontologia). Il prototipo risultante e' stato ispirato e largamente applicato al linguaggio naturale controllato per le specifiche dei dispositivi elettrici.

1. Introduzione

Il sempre crescente interesse verso metodologie di apprendimento a distanza (*e-learning*), legato più che alla necessità di alleggerire il carico didattico del personale

¹ Sviluppato all'interno della ricerca finanziata dal progetto nazionale "Web-Learning"

docente quanto ad una aumentata richiesta in tal senso da parte di studenti distanti dal luogo di insegnamento, richiede lo sviluppo di strumenti informatici di supporto differenziati adeguati alle esigenze sia dell'insegnamento che dell'apprendimento.

L'utilizzo di tecnologie informatiche in tale campo, pur conservando i limiti intrinseci di uno strumento automatico non in grado di sostituire pienamente il compito di un docente umano, può però portare numerosi vantaggi. Le metodologie di *e-learning* mirano a supportare, per loro stessa definizione, un percorso formativo personalizzato, asincrono e completamente *autogestito* da parte del discente, che è così in grado di sviluppare il percorso di studio al ritmo del proprio apprendimento. I supporti informatici possono inoltre consentire un apprendimento più veloce, grazie all'uso di tecnologie multimediali e di un elevato grado di interattività.

In tale contesto, il Sistema di Apprendimento di Linguaggio Naturale Controllato (SALINAC) presentato in questo lavoro, propone un approccio automatizzato all'apprendimento del Linguaggio Naturale Controllato (LNC). Lo scopo è quello di guidare l'utente all'apprendimento del LNC, attraverso una sottomissione di testi inizialmente in linguaggio naturale che siano successivamente sempre più aderenti alle specifiche ed ai vincoli del LNC.

La metodologia di apprendimento proposta prevede la scrittura, da parte del discente, di un testo all'interno di un editor (specificamente Microsoft Word®). Il testo viene *catturato* dalla componente *client* del sistema, ed inviato al *server*, dove viene effettuata l'analisi del testo inserito. Il *server* fornisce quindi i risultati di tale analisi al *client*, evidenziando errori di scrittura e suggerendo eventuali miglioramenti o correzioni apportabili al testo affinché aderisca maggiormente al LNC. Il *client* visualizza tali informazioni. Utilizzando tecniche di parsing robusto, di machine learning e di analisi sintattico-semantica, il sistema è in grado quindi di guidare gradualmente l'utente alla comprensione e all'acquisizione del LNC e dei suoi vincoli lessicali, sintattici e semantici².

Il sistema SALINAC è stato sviluppato in vista di una sua applicazione nell'ambito di uno specifico Linguaggio Naturale Controllato di dominio Elettrico (LNCE) sviluppato dal centro CESI dell'ENEL [Ciapessoni et al,2000a], [Ciapessoni et al,2000b], in collaborazione con l'ENEL. E' comunque prevista una sua applicazione in ambiti di dominio differente, attraverso l'adozione di lessici e di basi di conoscenza adeguate.

L'architettura di SALINAC, la sua modalità di interazione con l'utente e la sua struttura client-server possono altresì essere riutilizzate in altri ambiti di e-learning in cui siano previste modalità di apprendimento basate su piattaforme software di tipo testuale.

Nei paragrafi successivi vengono descritte dettagliatamente le funzionalità necessarie per la realizzare un sistema automatico di apprendimento di LNC (*paragrafo 2*) e l'architettura del sistema SALINAC (*paragrafo 3*), per concludere quindi con alcune considerazioni finali (*paragrafo 4*).

² Anche se applicata all'acquisizione di un LNC, la metodologia di apprendimento può essere considerata valida anche in un contesto più ampio e complesso di apprendimento di una lingua naturale.

2. Funzione di un sistema di supporto all'apprendimento di un LNC

Per favorire un efficace ed efficiente processo di comprensione del linguaggio, un sistema di supporto all'apprendimento di un LNC deve possedere funzionalità correlate sia ad aspetti di controllo ed analisi del testo, sia a proprietà di usabilità, robustezza e flessibilità.

Il sistema inizialmente deve lasciare all'utente una completa libertà espressiva nella scrittura del testo in linguaggio naturale, in modo da poter far fronte alle diverse successive esigenze e livelli di apprendimento e conoscenza del LNC da parte dell'utente stesso.

In secondo luogo il sistema deve essere in grado di:

- analizzare il testo dell'utente;
- verificarne l'aderenza alle specifiche ed alle caratteristiche del LNC, tanto dal punto di vista morfologico e grammaticale, che da quello sintattico e strutturale;
- proporre in maniera interattiva all'utente suggerimenti per la correzione del testo, esplicando in tal senso la sua funzione di supporto all'apprendimento.

La verifica a vari livelli della correttezza del testo e l'eventuale segnalazione di errori e suggerimenti, costituiscono quindi l'elemento fondante del sistema. La correttezza del testo deve essere verificata attraverso l'attinenza del testo a tre diverse tipologie di vincoli che possono essere imposti da un LNC:

- *vincoli di tipo ortografico e morfologico*: questi vincoli sono caratterizzati da regole di restrizione all'uso di parole la cui località si confina tipicamente ad una singola parola, come l'utilizzo di una terminologia tecnica e specifica dell'LNC definita in un dizionario;
- *vincoli di tipo sintagmatico*: vincoli che restringono l'uso di strutture frasali della lingua italiana mediante criteri la cui località coincide con un sintagma o con l'intera frase.
- *vincoli di tipo semantico*: vincoli relativi alla restrizione d'uso di costrutti sintagmatici, o all'uso di forme referenziali (ad es. anafora pronominale) o alla libertà di introdurre specificatori ad un livello arbitrario di annidamento. La *località* di queste norme quindi include il piano strutturale e al contempo quello delle interpretazioni semantiche. La realizzazione di modelli adeguati di tali vincoli risulta di difficile realizzazione, a causa della complessità intrinseca nel calcolo delle interpretazioni intese dallo scrivente ma non correttamente espresse in LNC.

La verifica dell'ultima tipologia di vincoli risulta particolarmente complessa a causa della carenza di un modello unificante ed esplicativo dei fenomeni semantici interni ad un documento, ed a causa della incompletezza delle informazioni a disposizione durante l'analisi del testo. Conseguentemente, la comprensione dell'interpretazione intesa dallo scrivente e non correttamente espressa nello LNC risulta difficoltosa, ed è quindi necessario affidarsi a modelli di vincoli in grado di gestire l'incertezza (interpretazioni multiple di una singola frase errata) e l'assenza di informazione.

A tale scopo è quindi utile ricorrere a strumenti robusti e flessibili, in grado di gestire situazioni critiche, incerte, e di difficile soluzione. Metodologie di *parsing robusto* e di *apprendimento automatico* (o *machine learning*) [Carbonell et al,1986]

sembrano poter rispondere adeguatamente a tali caratteristiche. In particolare le tecniche di *machine learning* possono consentire una soddisfacente gestione dell'incertezza nella scelta della ristrutturazione di un testo semanticamente errato, essendo in grado di fornire più soluzioni alternative, anche se incerte.

Inquadrato in ambito di *Machine Learning* la soluzione al problema deve prevedere la classificazione di alcune *istanze* (le porzioni di testo su cui effettuare la verifica di correttezza) in una o più classi (le *classi di trasformazione*, ovvero la tipologia di errore presente nel testo e la relativa correzione). E' necessario quindi innanzitutto rappresentare le porzioni di testo (*periodo*) in un formalismo che sia comprensibile ad un algoritmo di *machine learning*. Uno dei formalismi più indicati allo scopo è quello di tipo *vettore attributo-valore*. Si devono quindi esprimere le proprietà sintattico-morfologiche del periodo in un vettore di attributi utili per il compito di classificazione in esame e rappresentanti caratteristiche del testo strettamente legate ai vincoli imposti dal LNC. Il vettore può ad esempio catturare caratteristiche quali:

- il numero totale di proposizioni e di parole costituenti il periodo;
- il numero medio di modificatori dei ruoli sintattici del periodo;
- il numero di congiunzioni e disgiunzioni;
- le caratteristiche morfologiche dei verbi.

Una volta identificati il formalismo e gli attributi, devono essere individuate le *classi di trasformazione*, ovvero le tipologie di errore riscontrabili nel testo (cui è associata la relativa correzione). Se fornito preventivamente di un insieme di esempi di periodi errati cui è associata la classe di trasformazione (*training set*), l'algoritmo apprenderà il modello di classificazione adeguato. In base ad esso il learner (detto *structural checker*) sarà quindi in grado di capire se il testo introdotto dall'utente è errato, ed in tal caso individuerà una o più classi di errore più probabili, associando ad esse la ricostruzione corretta in LNC.

3. Architettura del Sistema

L'architettura del sistema realizzato tiene conto delle caratteristiche delineate nel precedente paragrafo. Il sistema *SALINAC*, attraverso l'utilizzo di una interfaccia intuitiva (*Figure 3 e 4*) e di sistemi efficaci di correzione e suggerimento, è in grado infatti di consentire un apprendimento veloce e puntuale del LNC da parte dell'utente.

In particolare *SALINAC* esplica le seguenti funzionalità:

- *Editing*: scrittura da parte dell'utente di documenti in linguaggio naturale (*editor*);
- *Parsing*: analisi grammaticale dei documenti in fase di scrittura (*parser*);
- *Analisi interna alle preposizioni (infraclausal analysis)*: analisi della adesione delle singole frasi ai criteri di LNC (*inner propositional checker*);
- *Analisi della struttura frasale*, che impone i vincoli di complessità strutturale alla intera frase (*structural checker*);
- *Supporto alle correzioni* (comunicazione e *re-editing steps*), che consente (all'interno dello stesso ambiente di *editing*) di recepire gli errori (comunicazione

dei tipi e della localizzazione delle strutture non accettabili in LNC) e di proporre le modifiche necessarie in modo il più possibile amichevole, *re-editing* (*answer generator*).

L'insieme di queste funzionalità consente all'utente di prendere coscienza dei propri errori in maniera interattiva, e di incrementare quindi la sua conoscenza del LNC grazie ai suggerimenti forniti dal sistema.

Lo schema architetturale di SALINAC, descritto in *Figura 1*, comprende due macro-componenti: il sottosistema *client*, ed il sottosistema *server*.

La componente *client side* include il sottosistema di *editing* e quello di interazione (modulo *client* in figura). Il primo sottosistema coincide con una piattaforma standard per la scrittura di documenti e si è deciso di utilizzare Microsoft Word per tali attività. La scelta è stata motivata dai principi di portabilità e immediato uso nell'applicazione. Questa decisione ha indotto una serie di altre scelte implementative, come ad es. la adozione di una componente di interazione basata sulla tecnologia degli agenti di Microsoft.

La componente *server* include il *parser*, l'*analizzatore interno delle proposizioni*, l'*analizzatore strutturale* ed infine il *sottosistema di generazione di risposte* (*answer generation*). Il *parser* coincide con il sistema CHAOS [Basili et al,1998][Basili et al,2000][Basili e Zanzotto,2002] e consente la derivazione dei grafi sintattici (XDG) a partire dalle frasi in fase di scrittura.

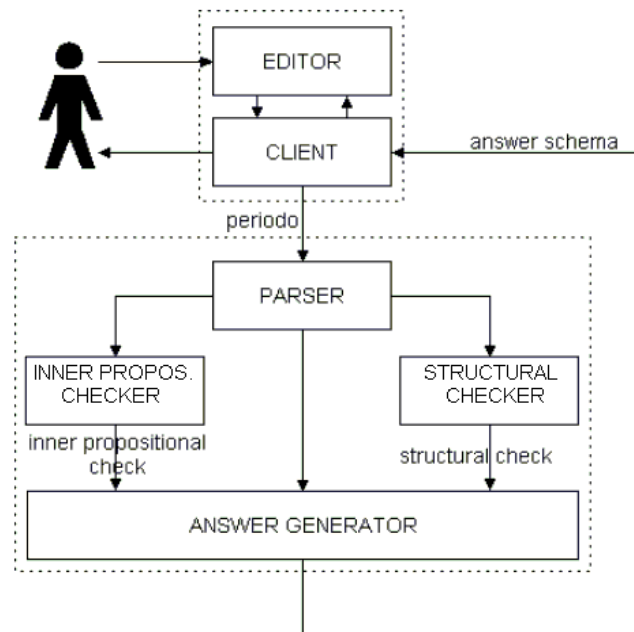


Figura 1. Schema architetturale del sistema SALINAC

I due analizzatori elaborano indipendentemente (seppur in cascata) i grafi sintattici: l'*analizzatore interno delle proposizioni* si occupa di verificare la adesione ai vincoli lessico-morfologici, sintattici e sintattico-semantici della frase in ingresso con una analisi il cui *scope* è quello della singola proposizione L'*analizzatore strutturale* coincide invece con la componente di classificazione del sistema di *machine*

learning. I due tipi di verifiche (*check*) forniscono poi l'input per il sottosistema di *answer generation*. Questo si occupa, a partire dalle analisi fatte, e quindi dai tipi di errori e dalle descrizioni dei segmenti errati, di costruire i diversi messaggi di interazione con l'utente per la localizzazione e la descrizione dell'errore ed il supporto alla sua rettifica.

3.1. Il parser

L'utilizzo di un parser adeguato alla particolarità del problema dell'apprendimento è di fondamentale importanza per una implementazione efficace del sistema. L'uso di tecniche di *parsing sintattico robusto* consente di ovviare ai problemi di categoricità (risponso booleano vero/falso del riconoscimento grammaticale) e di copertura (limiti nella gestione di fenomeni errati non previsti) degli approcci monolitici al *parsing*, che costituiscono gli elementi critici della metodologia di apprendimento adottata.

Si è quindi deciso di utilizzare un *parser robusto*, specificamente il sistema CHAOS, sviluppato dal gruppo di Ricerca dell'Università di Tor Vergata [Basili et al,1998][Basili et al,2000][Basili e Zanzotto,2002].

Il parser CHAOS tenta di cogliere i vantaggi sia delle metodologie di *parsing superficiale* (*shallow parsing*) che di *parsing lessicalizzato*, proponendo una architettura modulare che abbia come caratteristica principale la *robustezza*, ovvero la costanza delle prestazioni attraverso domini conoscitivi differenti. Questa flessibilità viene raggiunta attraverso l'integrazione di approcci *non pretenziosamente* lessicalizzati nell'ambito di piattaforme di parsing superficiale: l'architettura risultante, altamente modulare, permette quindi la creazione di analizzatori sintattici riconfigurabili che possano esibire un elevato grado di *robustezza*.

I moduli costituenti CHAOS comprendono:

- *Tokenizer*: ha lo scopo di trasformare il testo *pulito* in ingresso da una mera sequenza di caratteri ad una sequenza di parole.
- *Analizzatore Morfologico*: ad ogni parola viene associato il suo ruolo sintattico e le sue proprietà morfologiche, se necessario anche in maniera ambigua.
- *Yellow Page Tool*: ricerca nel testo i lemmi presenti in un elenco di parole conosciuto e caratteristico del dominio (modulo lessicalizzato).
- *POS Tagger* [Brill,1992]: effettua il Part-of-Speech tagging.
- *Named Entity Recognizer* [Pazienza e Vindigni, 2000]: individua nel testo nomi propri e strutture identificabili come Named Entity.
- *Chunker* [Abney,1996] [Basili et al,1998] [Federici et al,1996]: riceve in input le parole individuali, e genera sequenze di parole adiacenti come unità grammaticali, detti *chunk* (ad es, *sintagmi nominali* o *verbali*).
- *Verb Shallow Analyzer* [Basili et al,1998] [Basili et al, 2000] [Basili e Zanzotto, 2002]: analizza le strutture verbali sottostanti alle singole proposizioni nella frase. Tipici legami grammaticali derivati sono *soggetto* (SUBJ), *oggetto* (OBJ), e *modificatore verbale* (PPMOD).
- *Surface Shallow Parser* [Basili et al,1992]: individua all'interno del testo tutte le ulteriori dipendenze sintattiche (ad es. nomi-specificatori *scatto-di-interruttore*).

3.2. *L'analizzatore interno delle proposizioni*

Il modulo si occupa della analisi interna alle proposizioni costituenti una frase sottoposta al *check* di adesione ai principi di LNC. L'analizzatore esamina la struttura sintattica della frase fornita dal *parser*, ed in base ad essa esegue i controlli di adesione del testo ai vincoli dell'LNC. L'analisi si basa su lessici e ontologie specifiche di dominio, in modo da rendere l'architettura indipendente dalle restrizioni specifiche del linguaggio in questione e garantire un alto grado di riusabilità. I primi esprimono vincoli linguistici e le seconde definiscono proprietà generali del dominio essenziali alla validazione dei processi documentali. Di seguito verranno descritti i controlli sul LNCE³, realizzati nel sistema SALINAC in una sua implementazione.

Controllo dei complementi verbali. Svolge due principali attività. La prima consiste nel controllo sintattico della validità dei modificatori verbali introdotti da preposizioni, in base agli assunti sui vincoli alla modifica dei sintagmi verbali propri del LNC. La seconda consiste nel controllo semantico della validità delle interpretazioni dei modificatori preposizionali così come vincolate dal LNC. Nel caso infatti in cui sia possibile fornire una semantica per il verbo in esame allora il tipo di modificatore preposizionale potrebbe sottostare ad alcuni vincoli di natura semantica. Ad esempio un vincolo LNC potrebbe forzare particolari complementi ad essere utilizzati solo in associazione a sintagma nominali di un certo tipo. Nel caso particolare del LNCE sono stati catturati vincoli semantici riguardanti verbi che denotano concetti come *attività trasformativa*, *eventi trasformativi*, *stati permanenti o temporanei*. A questi verbi sono state quindi associati vincoli di tipo semantico che legano ad essi complementi di tipi particolari a seconda della preposizione.

Controllo dei legami soggetto. Nel caso del LNCE il controllo verifica che la presenza nel grafo sintattico *xdg* di una frase dei soggetti dei verbi sia completa e che non esistano subordinate senza soggetto (ad es. relative che mantengono un elevato grado di ambiguità).

Controllo dei modificatori dei sintagmi nominali. I modificatori dei sintagmi nominali vengono prodotti dall'analizzatore sintattico superficiale del *parser*, il quale individua le dipendenze di tipo non verbale dell'*xdg*. E' possibile che alcuni vincoli di LNC impongano restrizioni su questo tipo di strutture. La fase specifica di controllo dei modificatori dei sintagmi nominali serve esattamente a questo scopo ed impone alle strutture complesse di sintagmi nominali i vincoli necessari. Nel caso del LNCE, ad esempio, il controllo consiste nell'esaminare le strutture di link tra chunk nominali e chunk preposizionali, e nel rimuovere le interpretazioni in contrasto con determinati vincoli. In caso di errore il sistema fornisce l'insieme di tipi di errori associati al frammento in analisi e le possibilità di ricostruzione corretta, lasciando all'utente la scelta sulla modifica opportuna.

Controllo coordinative e subordinate. I vincoli LNC possono riguardare anche la struttura interna del periodo. Può ad esempio essere impedito da vincoli di LNC l'utilizzo di periodi particolarmente complessi o eccessivamente strutturati ed annidati, che risultino quindi di difficile od ambigua comprensione. Nel caso del

³ Ulteriori dettagli sull'LNCE possono essere trovati in [Ciapessoni et al, 2000a] [Ciapessoni et al, 2000b].

LNCE ad esempio viene limitato a due il numero di gradi di subordinazione massimo possibili in un periodo. Altri vincoli possono riguardare la regimentazione del modo in cui i differenti tipi di frasi subordinate possono essere espressi. In LNCE ad esempio non è ammesso l'utilizzo di subordinate implicite dirette, di proposizione sostantive o relative accessorie; è inoltre vincolato l'utilizzo di preposizioni particolari per introdurre proposizioni di un certo tipo (ad esempio le causali possono essere introdotte solo da *perchè*, e le finali da *affinché*).

3.3. L'analizzatore strutturale

Al fine di suggerire all'utente possibili correzioni alternative da apportare al testo nel caso di una sua organizzazione troppo complessa a livello strutturale, tanto da contravvenire ai vincoli del LNC, è stato realizzato un tool in grado di produrre tali alternative, partendo dal testo errato. Questo compito è riconducibile alla soluzione di un *problema di classificazione*, e quindi di Machine Learning, e la rappresentazione formale dei periodi adeguata è quella di tipo *vettore attributo-valore*.

Nel caso specifico del LNCE si è ricorso ad un algoritmo di apprendimento basato sull'utilizzo di due differenti algoritmi di Machine Learning, entrambi di tipo *decision-tree* (l'ID3 [Quinlan,1986][Quinlan,1996]). Il primo algoritmo (ML1) effettua la classificazione dei periodi in base alle loro caratteristiche generali; il secondo (ML2) in base alle caratteristiche delle singole proposizioni costituenti il periodo stesso. In tal modo possono essere più facilmente individuate le violazioni di regole LNC dovute a proprietà dell'intero periodo o delle singole proposizioni. Le decisioni di classificazione prese dai due algoritmi ML1 e ML2 vengono quindi armonizzate da un successivo modulo (il *voter*) che in base ad esse decide quale classe di errore assegnare al periodo. I vettori di ingresso di entrambi gli algoritmi prendono in considerazione caratteristiche morfologiche e sintattiche del testo (morfologia dei verbi, numero delle parole e delle frasi, numeri di modificatori, ecc.). Le classi di errore individuate per il LNCE hanno compreso errori che violano diversi vincoli del linguaggio controllato, tra cui;

- *vincoli di subordinata implicita diretta*: il testo dell'utente presenta una proposizione subordinata implicita diretta, il cui uso non è permesso in LNCE;
- *vincoli di relativa esplicita propria accessoria*: il testo dell'utente presenta una proposizione relativa esplicita propria accessoria, non ammessa in LNCE;
- *vincoli di indipendenti coordinate*: l'utente ha utilizzato due proposizioni indipendenti coordinate invece di proposizioni indipendenti autonome, come richiesto dal LNCE.

3.4. Il generatore di risposte

Il generatore delle risposte assume in ingresso il risultato delle analisi del modulo di analisi interna delle proposizioni e del modulo di analisi strutturale e genera il flusso di informazioni che la componente *client* richiede per la gestione dell'interazione con l'utente finale. Esso rappresenta quindi un componente critico dell'architettura del sistema: da una buona realizzazione del generatore dipende infatti l'espressività e la completezza delle informazioni prodotte dal sistema di analisi del testo, e quindi la qualità dello strumento.

Analizziamo lo schema che la risposta assume in generale, nel caso non banale in cui è stato individuato almeno un frammento errato. Lo scopo della risposta è quello di fornire allo scrivente tre tipi di informazioni:

- individuare i frammenti di testo in ingresso che contengono potenziali errori di scrittura, cioè fenomeni grammaticali non legali nel LNC;
- per ognuno dei frammenti individuati generare una o più classi per l'errore sottostante;
- fornire suggerimenti per poter rimuovere gli errori.

Queste tre attività sono finalizzate a costruire lo *schema della risposta* che segue la struttura seguente.

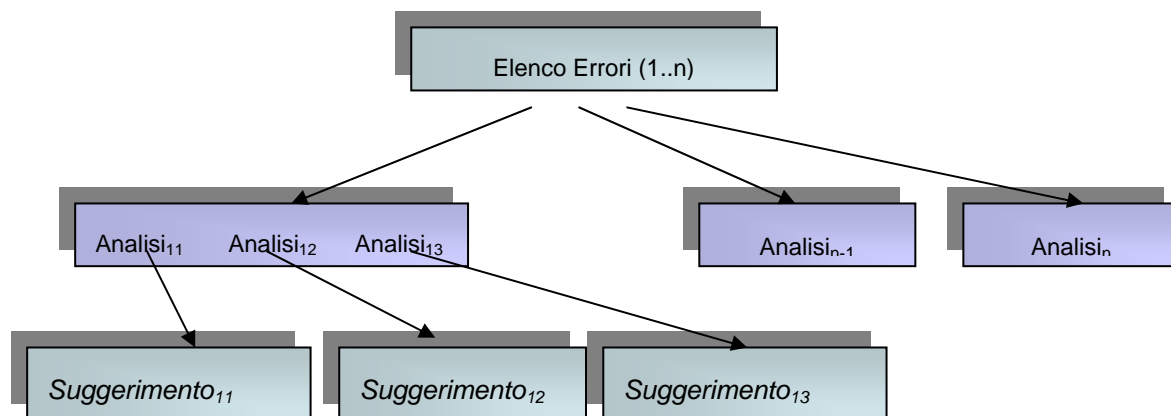


Figura 2. Schema della risposta prodotto dal generatore di risposte.

Elenco Errori è il menu iniziale di risposta che suggerisce l'elenco ((1..n)) dei diversi errori: questo elenco contiene i frammenti di testo in cui l'errore individuato si manifesta e guida l'utente nella ricerca della possibile correzione. Al primo livello, i menu riportano per ogni errore i -esimo le analisi possibili. Poiché in generale esiste più di una analisi per l'errore i l'elenco dei menù di primo livello riporterà una etichetta (classe possibile dell'errore) per ognuna. Alla selezione di una analisi j il menu attivato al secondo livello sarà quello relativo al suggerimento ij . Quando il suggerimento consiste di una riformulazione locale del testo, il modulo di generazione della risposta dato l'errore ed il tipo di analisi suggerite dai precedenti moduli del server (per es. dall'analizzatore locale di proposizioni) costruisce un'ipotesi per il testo di correzione. Tale testo viene fornito all'utente finale come il contenuto esplicito del menu di secondo livello corrispondente. Quando la ristrutturazione prevede una complessa analisi del testo sorgente l'errore e il tipo corrispondente di analisi vengono riportati come messaggio generale, esemplificato ma non direttamente dipendente dal testo in ingresso.

Esempi di schemi di risposta vengono forniti in *Figura 3 e 4*.

3.5. Il sottosistema client

Il sottosistema *client* ha il compito di presentare all'utente i risultati dell'analisi effettuata dal *server* sul testo da lui introdotto nel *text editor* di *Microsoft Word*®. Dovendo quindi rendere espliciti all'utente i risultati dell'analisi di attinenza al LNC del

documento, il *client* costituisce il *tutor virtuale*, ovvero il mezzo attraverso il quale l'utente può apprendere il LNC, imparando dai suoi errori e dai suggerimenti correttivi proposti dal *server*.

Grazie all'utilizzo di Visual Basic (VB) come ambiente di programmazione, si è realizzata la funzionalità *client* come una libreria dinamica (*dll*) del S.O. Windows con indubbi vantaggi per l'efficienza, ed eleganza del servizio. In tal modo è stato quindi possibile aggiungere alla interfaccia utente di Microsoft Word© una nuova barra degli strumenti, che contiene i comandi relativi alle funzioni *client*, quali invocazione e interazione con l'agente di correzione. Per l'utente si tratta quindi di utilizzare il sistema di *editing* con l'abituale facilità d'uso, avendo come unica cura l'attivazione del server, che rimane in ascolto durante la sessione lavorativa.

Il sottosistema *client* permette quindi l'identificazione degli errori, la loro visualizzazione, ed i suggerimenti di ricostruzione, secondo lo schema indicato in *Figura 2*. L'utente è messo in grado di navigare nello *schema della risposta* attraverso un semplice ed intuitivo sistema di menù visuali a scelte alternative, e di recepire i suggerimenti all'errore introdotto, ed eventualmente effettuare la correzione.

Per rendere il processo di apprendimento più intuitivo e stimolante, il sistema *client* viene *impersonato* visualmente dal personaggio animato *Microsoft Agent Merlin*, che ha il compito di mostrare la messaggistica di interazione (comprendente lo schema di risposta prodotto dal server).

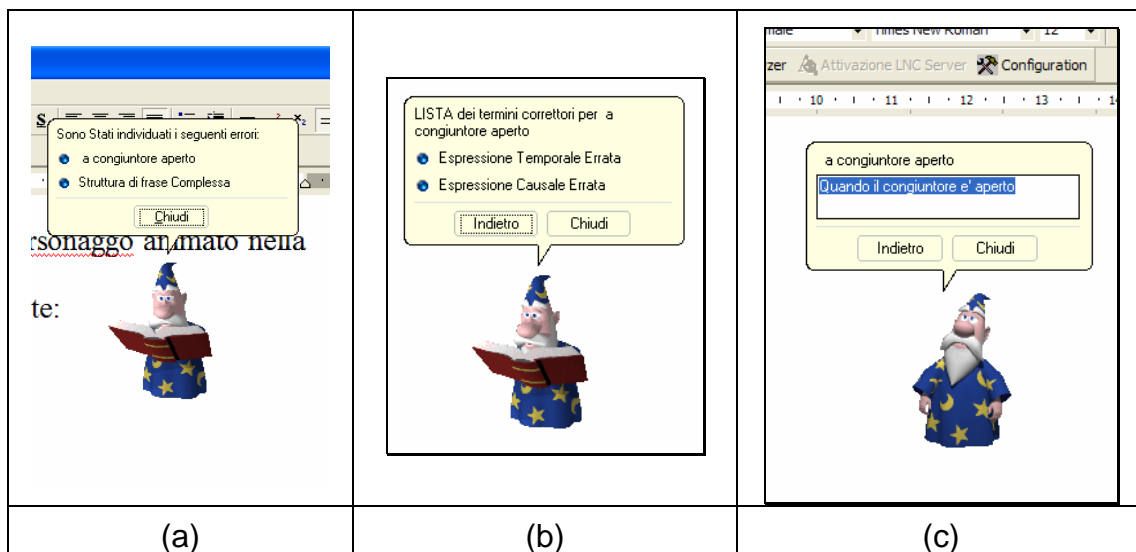


Figura 3. Esempio di interazione sistema-utente per l'LNCE.

Un semplice esempio di interazione tra *client* (Merlin) e utente riferito ad una frase errata in LNCE è descritta in *Figura 3*. Nella frase introdotta dall'utente:

“A congiuntore aperto le linee MT prese in considerazione sono solamente quelle che partono dalla sbarra relativa al trasformatore con protezione omopolare 59Vo avviata, mentre a congiuntore chiuso sono considerate tutte le linee MT di cabina.”

Il sistema rileva due errori (*Figura 3.a*), propone per il primo errore due analisi correttive (*Figura 3.b*), ed indica per la prima analisi una possibile ricostruzione corretta (*Figura 3.c*).

L' esempio seguente mostra in maniera più evidente le diverse tipologie di errore riscontrabili, ed i relativi suggerimenti di ricostruzione frasale (*Figura 4*). Si ammetta la frase in LNCE inserita dall' utente:

“Un avviamento della protezione di massima tensione omopolare sul j-esimo trasformatore è presente da un cane.”

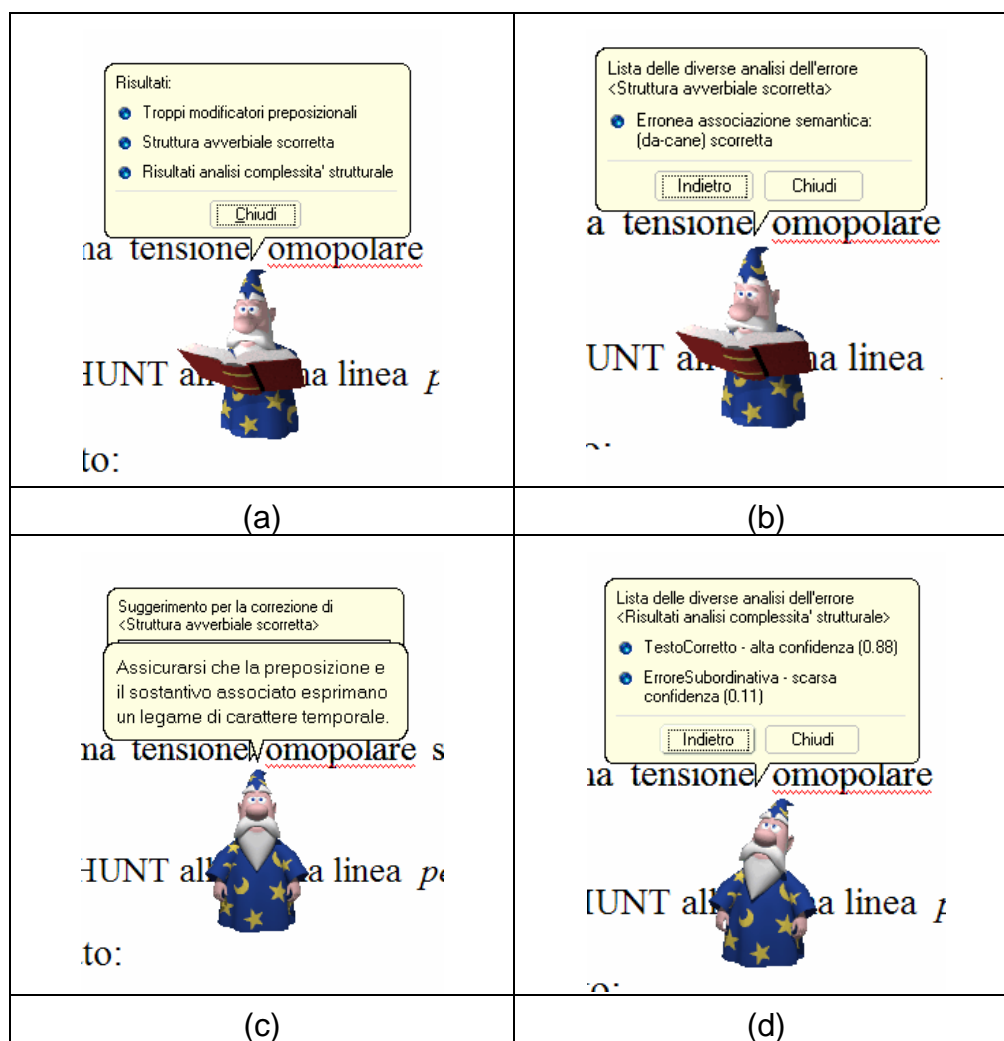


Figura 4. Esempio di rilevazione di errori per l'LNCE.

Il sistema rileva tre errori (*Figura 4.a*). I primi due errori vengono rilevati dall'*analizzatore interno delle proposizioni*, e sono di natura sintattico-semantica. In particolare il secondo errore rileva una struttura avverbiale di durata scorretta. L'utente può quindi analizzare più nel dettaglio l'errore navigando il menù (*Figura 4.b*): l'analisi proposta dal sistema rivela che l'associazione semantica tra la preposizione *da* ed il modificatore *cane*, non è di tipo temporale, come invece richiesto dai vincoli LNCE per strutture introdotte dalla preposizione *da*. L'utente può

quindi ottenere il suggerimento di ricostruzione (*Figura 4.c*) scendendo ulteriormente nei livelli del menù.

Il terzo tipo di errore viene rilevato dall'*analizzatore strutturale*, il quale indica all'utente (*Figura 4.d*) un possibile errore nella struttura del periodo. Più in particolare il sistema ritiene il testo dell'utente corretto dal punto di vista strutturale con un margine di confidenza dell' 88%, ma segnala altresì la scarsa probabilità che sia stato introdotto un errore di tipo subordinativo (che sia cioè presente nel periodo una frase subordinata implicita diretta, il cui uso non è permesso in LNCE).

4. Conclusioni

In questo lavoro è stato presentata SALINAC, una piattaforma software per il supporto all'apprendimento di linguaggi naturali controllati. La piattaforma consente il controllo intelligente delle attività di scrittura favorendo un processo di apprendimento del tipo *learning by doing* [Gibbs, 1988]. Il sistema applica tecniche robuste di elaborazione del linguaggio naturale e si basa su una rappresentazione esplicita dei vincoli linguistica (lessici morfologici e sintattica) e del dominio applicativo. Il prototipo risultante è stato ispirato e largamente applicato al linguaggio naturale controllato per le specifiche dei dispositivi elettrici.

Il sistema SALINAC sfrutta una architettura *client-server* consentendo una netta separazione tra i diversi tipo di validazione:

- sintattico-semantica, mediante la analisi semantica delle strutture sintagmatiche estratte da un *parser* robusto (CHAOS)
- strutturale, mediante l'indagine sulle strutture preposizionali interne alle frasi. La validazione procede qui in termini di classificazione delle strutture in classi di errore.

Sebbene il prototipo corrente sia applicato nell'ambito di uno specifico Linguaggio Naturale Controllato (dominio Elettrico (LNCE), in collaborazione con il Centro di Ricerca ENEL: CESI), esso rappresenta un approccio indipendente dal dominio, essendo l'interoperabilità semantica garantita da un approccio *ontology-based*.

L'architettura di SALINAC, la sua modalità di interazione con l'utente e la sua struttura client-server possono inoltre essere interamente riutilizzate in altri ambiti di e-learning in cui l'apprendimento coinvolga informazione di tipo testuale.

5. BIBLIOGRAFIA

Abney S., Partial Parsing via Finite-State Cascades, in Proceedings of the Eight European Summer School in Logic, Language and Information, Prague, Czech Republic, 1996.

Basili R., Pazienza M.T., Velardi P., A shallow Syntactic Analyzer to Extract Word Association from Corpora, in Literary and Linguistic Computing, 7, 114-124, 1992.

Basili R., Pazienza M.T., Zanzotto F.M., A Robust Parser for Information Extraction, in Proceedings of the European Conference on Artificial Intelligence (ECAI '98), Brighton (UK), 1998.

Basili R., Pazienza M.T., Zanzotto F.M., Customizable Modular Lexicalized Parsing, in Proceedings of the 6th International Workshop on Parsing Technology, Trento (Italy), 2000.

Basili R., Zanzotto F.M., Parsing Engineering and Empirical Robustness, Journal of Language Engineering, Cambridge University Press, 2002.

Brill E., A simple rule-based part-of-speech tagger, in Proceedings of {ANLP}-92, 3rd Conference on Applied Natural Language Processing, Trento (Italy), 1992, pp. 152-155.

Carbonell J.G., Michalski R.S. and Mitchell T.M., Machine Learning - An Artificial Intelligence Approach, Morgan Kaufman Publishers, Inc. Los Altos, CA, 1986.

Ciapessoni E., Bertani A., Castelnovo W., Botti O., Linguaggio naturale controllato per la specifica dei sistemi di automazione: definizione, CESI n. TEC-A0/022641, 2000 (a).

Ciapessoni E., Bertani A., Castelnovo W., Il linguaggio naturale controllato nella specifica dei sistemi di automazione, CESI n. 91/00014-02, 2000 (b).

Federici S., Montemagni S., Pirrelli V., Shallow Parsing and Text Chunking: a View on Underspecification in Syntax, in Proceedings of the Eight European Summer School in Lgic, Language and Information, Prague, Czech Republic, 1996.

Gibbs G., Learning by Doing: a guide to teaching and learning methods, Further Education Unit, London, 1988.

Pazienza M.T. , Vindigni M., Identification and Classification of Italian Complex Proper Names, International Conference on Artificial and Computational Intelligence For Decision, Control and Automation In Engineering and Industrial Applications (ACIDCA'2000), Monastir (Tunisia), 2000.

Quinlan J. R., Induction of decision trees, In Machine Learning Vol.1, 1986.

Quinlan J. R., Bagging, boosting, and C4.5. Unpublished manuscript, 1996.