C02 - AUTOMATIC EXPLORER

Qual è lo scopo della sonda?
Esplorare una porzione del territorio, senza fare ipotesi sull'ampiezza del pianeta.

Si da partenza e arrivo?
La specifica da un'indicazione che non deve calcolare il percorso: la sonda deve decidere gli spostamenti in base agli ostacoli che incontra.

Le informazioni morfologiche arrivano solo dalla sonda?
Come requisito opzionale è possibile scattare ed inviare foto a terra fatte direttamente dalla nave spaziale. Prima di tutto, va pensato se è questo aspetto è utile e fattibile rispetto alle nostre capacità. Si tenga inoltre conto che non sempre è possibile fare questo genere di attività, sopratutto su pianeti tipo Venere, ricoperti completamente da nubi gassose che impediscono la visione del suolo.

C03 - SQL Plugout

Codice dichiarativo SQL?
Non è richiesto codice esecutivo, praticamente solo il DDL.

Prevede che l'utente abbia un DBMS installato sul computer?
La presenza di un DBMS aiuta; quando si ha un diagramma delle classi si usa il plugout per aumentare la velocità di programmazione, ma serve qualcosa per provarle. Assumere di non averlo è ingenuo.

Per la rappresentazione del database c'è qualche standard da seguire?
Su questo il capitolato è molto vago, ci si aspetta che venga fatta investigazione sulle soluzioni esistenti. Si è scelto questo linguaggio perché si suppone che la soluzione esista già, ma non per il particolare strumento UML.

C05 – AEG: “Validazione timbrature ad alte prestazioni ed affidabilità”

Per informazioni: cossutti@antiblema.it

Lo scopo è quello di validare le timbrature dei dipendenti di un grosso ente pubblico controllando se gli ingressi e le uscite sono conformi ai contratti dei singoli dipendenti. Ogni dipendente è quindi in possesso di un tesserino magnetico che registra gli ingressi e le uscite.
Esistono già dei sistemi tuttora utilizzati tipo HR Systems. L'approccio tradizionale è con RDBMS che attraverso strutture dati specifiche vengono elaborati attraverso procedure.
Sono prodotti scarsamente ricorsivi: milioni di specifiche Cobol con tanti “if then else” legati alla complessità della dinamica del mondo del lavoro. Deve garantire la possibilità, in caso di cambiamenti di retribuzione, di capire a quanto ammonterebbe la retribuzione se si fosse rimasti al vecchio contratto. In altre parole ci possono essere variazioni contrattuali a ritroso nel tempo, anche di 2 anni indietro.

Processo di validazione:
Il dipendente passa il badge e lascia il log di quando e dove è entrato e lo stesso al momento

dell'uscita.

Errori del dipendente e gestione: le tessere hanno due sensi di timbratura, uno per l'entrata ed uno per l'uscita. Il sistema deve poter gestire questi errori, e capire che se un OUT precede un IN, l'utente ha commesso un errore o per aver timbratura male o per una dimenticanza.
Cambi contrattuali: un impiegato di primo livello viene passato a dirigente con due anni di retroattività.

Quello che vogliamo è passare ad un sistema nuovo:
Usare motori inferenziali: Impiego di **Drools** che attraverso un linguaggio dichiarativo che attraverso varie sentences modula la condizione contrattuale.
Grid: in momenti di picco (es: fine mese) l'elaborazione dei dati è parallelizzata su più macchine (parallel computing).
C'è un limite temporale per legge entro il quale il sistema deve garantire la risposta, compresa l'attività di rettifica da parte dei dipendenti.

Prodotto **Gigaspaces** che si basa su openspaces => scalabilità
ci sono n macchine virtuali che si distribuiscono il carico di lavoro e garantiscono il funzionamento anche in caso di caduta di una parte delle macchine.

Acquisire le timbrature da oracle.
Attraverso il motore di quadratura fare i conti.

Ci sono n istanze di jvm partizionate e distribuite all'interno del cloud (insieme di dati e messaggi). Modello di interoperazione tra le jvm. Space bus tiene sincronizzate le istanze durante la fase operativa.

Drools: Sistema di inferenza che valida i fatti sulle regole una sentence per volta.
Forward Chaining: è un motore che ci basa su regole ed è in grado di riprodurre le azioni elementari dato un risultato. (Es. voglio una Audi con cilindrata X, cambio automatico, al costo di Z, è in grado di ritornarmi tutte le auto che rispettano i vincoli ed illustrandomi la configurazione rimanente) Noi vogliamo usare questo motore per sapere se il dipendente è in regola per esser retribuito.

Considerazioni del Prof. Vardanega: scarso impegno sul piano algoritmico, grandi difficoltà tecnologiche.
Senza alcun dubbio la piattaforma di sviluppo dovrà esser java trasformando però le condizioni “if then else” in sentences.
Il codice si limiterà a costruire le interfacce di alimentazione della griglia e al codice di validazione.
Quando il sistema è consistente fare esercizi di verifica della scalabilità del sistema.

Vanno fatte anche elaborazioni di tipo statistico?
No, viene usato solo per calcolare gli stipendi.

L'elaborazione va fatta sui dati degli enti?
Ci sarà a disposizione una struttura dati contenente un subset dei dati forniti dagli enti.
Questi dati sono centralizzati a Padova.

Ci verrà fornita la lista dei vincoli? Serve interfaccia grafica?

Le definizioni dei vincoli sono già presenti nel sistema HR. Saranno fatti incontri col personale delle amministrazioni.
Ci sarà una un applicativo web, ma non è l'obbiettivo, drools fornisce già un supporto web per la generazione delle regole.

C04 – Ajaxdraw

Per informazioni: gregorio.piccoli@zucchetti.it
stefano.ceschiberrini@zucchetti.it
alessio.rambaldi@zucchetti.it

Zucchetti è un'importante software house italiana, copre procedure gestionali e contabilità per le aziende. Programmi di gestione dello studio dei commercialisti.
Lo scopo dell'appalto è realizzare un disegnatore di figure vettoriali in ambiente totalmente web.
Nel linguaggio a markup HTML5 c'è un nuovo tag, **canvas**, che permette di fare disegni in modo totalmente libero. Si deve quindi sfruttare tale tag per fare un classico programma di disegno che crea e permetta di salvare immagini vettoriali con estensione svg.

Problema di compatibilità: implementazioni leggermente differenti tra i browser. Firefox, Safari, Chrome ed Opera implementano il linguaggio HTML5. Internet Explorer (comunque **opzionale**) no ma c'è un modo per scavalcare il problema cioè **excanvas**.
La compatibilità dell'applicazione deve essere solo con le versioni dei browser che forniscono il tag canvas, non precedenti.

Perchè proporre questo tipo di progetto?
Perchè sono interessati ad un proof of concept.

Libera scelta delle librerie.

Problemi prestazionali:
Inkscape è scritto in c++ e su documenti con vari oggetti è molto lento. Come si potrebbe comportare uno con motore javascript che disegna su un browser?
L'ipotesi è che da questo disegnatore, in un futuro, si possano ricavare dei disegnatori specializzati.
Non è concepibile la concorrenza con photoshop.
L'idea è che i disegnatori specializzati non abbiano bisogno di prestazioni, ma hanno grande valore.
C'è da dire comunque che per fare cose semplici è sicuramente più veloce il canvas che inkscape.

Deve esser gestire anche grafica bitmap? No, solo svg.

Salvataggio?
Lo script non può scrivere in locale. Proposta google gears o salvataggio su server.

C01 – GOGOL

Lo scopo è costruire un sistema software per la generazione, la gestione e l'ottimizzazione dell'orario delle lezioni di un corso di studi universitario, comprensivo di un ambiente configurabile tramite interfaccia web accedibile tramite autenticazione.

Utenti:
- Presidente del CCS
- Personale della segreteria
- Docenti coinvolti

Non deve esserci sovrapposizione tra i corsi di anni adiacenti.
Le ore sono da 60 minuti ma è preferibile che sia un'informazione decidibile dall'utente.
È preferibile che le ore dello stesso corso siano 2 adiacenti.
La segreteria rende disponibili le aule e i docenti del corso di laurea in informatica e probabilmente anche di altri corsi di laurea.

Funzionalità a disposizione degli utenti:
- Inserire dati del corso di laurea
- Inserire di vincoli da parte dei docenti
- Inserire vincoli riguardanti la struttura dell'orario
  come la ripartizione dei corsi sui vari anni e periodi
- Generare dell'orario di un trimestre
- Generare suggerimenti per eventuali inconsistenze
- Aggiornare l'orario in caso di modifiche nell'orario

Vincoli e preferenze (i vincoli vanno soddisfatti, le preferenze il più possibile):
se tutto viene considerato come vincoli, difficilmente si trova una soluzione.

Dati del corso di laurea:
- Anno (I - II - III - I magistrale - II magistrale):
  Corsi dello stesso anno non si devono sovrapporre.
- Periodo: per ogni trimestre si fa il calcolo.
- Nome/i docente/i: si devono ovviamente evitare le sovrapposizioni
- Stato (obbligatorio/opzionale) minimizzando le sovrapposizioni
- Ore previste di laboratorio ed in aula
- Stima del numero di studenti per decidere le aule

Inserimento struttura orario (segreteria didattica):
- Anno accademico
- Periodi di lezione, giorni di vacanza ecc.
- I anno mattina, II anno pomeriggio, III anno mattina (vincolo/preferenza)
- Buchi più piccoli possibili (orario il più compatto possibile) (preferenza)
- Ore da occupare durante la giornata
- Lezioni di 2 ore (preferenza)

Inserimento dati aule (segreteria didattica):
- Aule disponibili
- Capienza (vincolo)
- Orario e periodi di non disponibilità (vincolo)

Inserimento dati docente (singoli docenti):

- Giorni e ore di indisponibilità con motivazione:
  il docente non più avere lezioni in quei giorni/ore (vincolo)
- Preferenze su orari e giorni con motivazione
- L'inserimento e l'accesso accesso devono avere periodi limitati

Per ogni periodo, **generare l'orario** con tutti i corsi del periodo, **tale che:**

- Tutti i vincoli siano soddisfatti
- Le preferenze siano soddisfatte il più possibile
  se non è possibile soddisfare tutti i vincoli, il sistema deve suggerire vincoli da rilassare
  o proporre soluzioni sub-ottime

L'**interfaccia** deve essere **web** con form per inserire i dati, preferibilmente con menù a tendina per evitare inserimenti errati.

L'accesso degli utenti deve prevedere **diritti diversi:**

- Sempre e dovunque la segreteria didattica ed il presidente CCS
  ed inoltre deve gestire gli account
- Solo sui propri dati e in certi periodi ai singoli docenti

La base di dati deve contenere tutti i dati: corsi, docenti, corso di laurea ...
L'inserimento può esser fatto anche in più fasi, deve prevedere la correzione di eventuali dati errati e l'aggiunta di vincoli ad una soluzione proposta.

Output: orario in formato PDF e/o HTML con formattazione facilmente leggibile.

Come generare un orario?

Due metodi principali di ricerca nello spazio delle soluzioni

- **Ricerche sistematiche ed euristiche:**
  Ricerca l'ottimo localmente, costruisce una soluzione iniziando da una soluzione a caso, ad ogni passo, effettua una piccola modifica della soluzione corrente, in modo da ottenere una soluzione migliore, ad esempio modificando il valore di una sola variabile.
  Si ferma quando non c'è modo (tra quelli previsti) di migliorare.
  Può quindi non arrivare ad una soluzione ottima.
- **Sistema branch and bound:**
  costruisce una soluzione settando una variabile alla volta tipo numero di preferenze soddisfatte, somma pesata sulle preferenze. Ogni soluzione parziale viene confrontata con la ottima trovata finora, e si procede solo se c'è speranza di trovare una soluzione migliore. In caso di non miglioramento si ritratta l'ultima decisione, e si prova un altro settaggio per l'ultima variabile.
  L'algoritmo arriva sempre ad una soluzione ottima.

Dove deve girare?

È plausibile che il sistema funzioni con i server gestiti dai tecnici di facoltà. Libera la scelta dell'algoritmo e delle tecnologie. Requisito importante è che il software sia portabile.

Il problema algoritmico è abbastanza complesso, i programmi esistenti non funzionano bene.

Il software è realmente fattibile?

Guardare il software esistente, provare a fare la scelta migliore ed ammettere che il software sia modificabile e quindi non totalmente automatizzato.

Gestire anche la magistrale?

La magistrale ha i vincoli aggiuntivi di indirizzo. Si può pensare al problema generico comprensivo di indirizzo, che non aumenta la complessità.

Riguardo alle tempistiche?

La generazione dell'orario non deve prevedere risposte istantanee ma in tempi ridotti, al più un'ora.

Non bisogna schedulare un trimestre, ma una settimana.

Chi gestisce gli account?

C'è un amministratore che gestisce gli accessi corretti. Solo deve essere configurabile.

Interoperabilità dei dati?

Il prototipo deve essere capace di poter utilizzare un qualsiasi formato. Ad esempio input da xml e output su xml che non dipende dalla tecnologia realizzativa.

Per l'integrità dei dati è necessario fare backup? Ridondanza?

Certo. Immettere i dati iniziali è l'operazione più lunga. Si vuole fare una sola volta. Sarebbe inaccettabile perdere il database. Deve essere garantita la durabilità dei dati.

Ambiente di prova con dati pre-caricati per far vedere che il tutto funziona. Anche filmati già fatti con prove già svolte. Qualcosa che aiuti le persone a capire quello che devono fare.

Scriba: Alessandro Bruni

Integrazione: Angelo De Menis