Ristrutturazione del Class Diagram

# 1.Analisi delle ridondanze

Non compaiono evidenti ridondanze nel class diagram.

# 2.Analisi delle chiavi primarie

A eccezione delle entità **Progetto**, **Dipendente** e **SalaRiunione** che presentano le loro chiavi primarie composte rispettivamente da **CodProgetto** (un codice seriale generato a partire da alcune informazioni sul progetto come l’ambito e il tipo di ricerca), **CF** (il codice fiscale del dipendente generato a partire dai suoi dati anagrafici) e **CodSala** (sigla della sala composta di pochi caratteri), le altre classi già possiedono chiavi primarie artificiali composte da soli valori interi unici (es. **IDMeeting**, **IDSkill**).

# 3.Analisi degli attributi multipli

La classe **Progetto** presenta un attributo multiplo **AmbitoProgetto** perché uno stesso progetto può avere più ambiti di applicazione. Dato che il modello di dati relazionale non prevede questo tipo di attributi, abbiamo creato una classe **AmbitoProgetto** che opportunamente associato alla classe **Progetto** sostituisce l’attributo. In questo modo possiamo evitare di imporre un limite sul numero di ambiti associati a un progetto.

La classe **Dipendente** presenta un attributo multiplo **Telefono** inerente ai possibili contatti telefonici inseriti dal dipendente. Considerando che in media una persona comune può essere contattata con solo due numeri telefonici (uno di casa e uno del cellulare personale), abbiamo pensato fosse sufficiente offrire due attributi singoli parziali **TelefonoCasa** e **Cellulare** in sostituzione di **Telefono**.

# 4.Analisi delle classi di associazione

L’unica classe d’associazione presente nel class diagram è **RuoloDipendente** che vincola un dipendente ad assumere un ruolo specifico quando partecipa a un progetto. La soluzione migliore è sostituire questa classe di associazione con una classe vera e propria **RuoloDipendente** opportunamente associata sia a **Progetto** che a **Dipendente**.

# 5.Analisi delle gerarchie

Il class diagram non presenta specializzazioni e generalizzazioni.

# 6.Analisi dei dati strutturati

La classe **Dipendente** presenta l’attributo strutturato **LuogoNascita** che va scomposto in qualche modo. Dato che l’intento del software è quello di autogenerare il codice fiscale del dipendente a partire dai dati anagrafici inseriti, preferiamo avere delle tabelle aggiuntive **Comune** e **Provincia** in cui sono memorizzati i comuni e le province italiane con annessi codici e associarle alla classe **Dipendente** opportunamente per ottenere le stesse informazioni senza l’attributo strutturato.