CONGRESO INTERNACIONAL DE
INNOVACIÓN DOCENTE E INVESTIGACIÓN EN EDUCACIÓN SUPERIOR
DESAFÍOS DE LA ENSEÑANZA Y APRENDIZAJE EN EDUCACIÓN SUPERIOR

ADQUISICIÓN DE COMPETENCIAS EN MÉTODOS FORMALES CON ESPECIFICACIONES EJECUTABLES

José Luis Sierra Rodríguez, Mercedes Gómez Albarrán, Ana María González de Miguel, Marta López Fernández
Dpto. Ingeniería del Software e Inteligencia Artificial. Facultad de Informática. Universidad Complutense de Madrid

OBJETIVOS

- Solución para mejorar la adquisición de competencias en especificaciones formales en programación.
- Analizar la integración de la solución en materias de algoritmia de titulaciones superiores en informática
- Evaluar la usabilidad de la solución.

MÉTODO

Basado en la "Investigación Activa" (*Action Research*):

- Análisis del estado del arte.
- Diseño y construcción de soluciones.
- Evaluación mediante experiencias.
- Difusión de resultados.

RESULTADOS

- Desarrollo de **ESPECIFICA++**: lenguaje integrado en C++ que permite construir especificaciones ejecutables.
- Integración de actividades **ESPECIFICA++** en jueces automáticos "en línea" (p.e., DomJudge, ampliamente utilizado en concursos de programación a nivel nacional e internacional).
- Evaluación de la usabilidad con casos de estudio, y análisis DAFO.

```
int max_len_cons_seq(int a[], int n) {
  /*PREC:*/assert(0 < n && n <= N);

   int i = 1;
   int len = 1;
   int resul = 1;

   /*INV:*/ assert(max_lcs(a, i, resul)());
   /*INV:*/ assert(last_cons(a, i, len)());
   /*INV:*/ assert(0<i<=n);
   /*BOUND:*/  assert((n - i) >= 0);
    while (i < n) {
     if (a[i] == a[i - len]) {
         i++;
         len++;
         if (len > resul) resul = len;
      }
      else { i++; len = 0;}

      /*INV:*/ assert(max_lcs(a, i, resul)());
      /*INV:*/ assert(last_cons(a, i, len)());
      /*INV:*/ assert(0<i<=n);
      /*BOUND:*/  assert((n - i) >= 0);
    }
    /*POST:*/assert(max_lcs(a, n, resul)());
    return resul;
}
```

Ejemplo de función C++ con anotaciones escritas en **ESPECIFICA++** (en texto resaltado)

```
auto max_lcs(int a[], int n, int resul){
  var(int, i);
  var(int, j);
 /* max i,j:0≤i≤j≤n ∧ cons_seg(a,i,j):(j-i)+1 */
  return
    exp(resul) ==
    maxi<int, int>(i, j).
      range(0, n - 1).
      filter(pred(*i <= *j)&&cons_seg(a,i,j)).
      val(exp((*j - *i) + 1));
}
```

Definición del predicado **max_lcs**

```
auto cons_seg (int a[], Var<int> i, Var<int> j){
 var(int, k);
 /* ∀k:i≤k≤j:a[i]=a[k] */
 return forall<int>(k).range(i,j).
            yield(pred(a[*i]==a[*k]));
}
```

Definición del predicado **cons_seq**

```
auto last_cons(int a[], int i, int len) {
  var(int, k);
/* (∀k:i≤k≤j:a[i-len]=a[k])∧
   (i-len =0 ∨ a[i-len-1] ≠ a[i-len] )   */
  return forall<int>(k).range(i-len, i-1).
           yield(pred(a[i-len] == a[*k])) &&
           pred(i-len == 0||a[i-len-1]!=a[i-len]);
}
```

Definición del predicado **last_cons**

La integración de **ESPECIFICA++** con C++ permite utilizar funciones C++ como mecanismo de abstracción durante el proceso de especificación

Especificación ejecutable **ESPECIFICA ++** … y dicha especificación expresada con la notación habitual

CONCLUSIONES

- **ESPECIFICA++** permite a los estudiantes ver las especificaciones como objetos activos que pueden ejecutarse, facilitando el aprendizaje de especificaciones formales.
- Las evaluaciones preliminares confirman su efectividad en cursos introductorios de algoritmia.
- Las debilidades identificadas incluyen la dependencia de C++, un diagnóstico de errores mejorable, y problemas de eficiencia.
- Actualmente, estamos trabajando en los aspectos a mejorar, en su portabilidad a otros lenguajes (p.e., Java) y en su implantación en la Universidad Complutense de Madrid.

Trabajo soportado por el programa **UCM INNOVA 2024-2025** (**proyecto 152**), y por el proyecto **PID2021-123048NB-I00**. Agradecemos a Antonio Sarasa Cabezuelo su participación en esta línea de investigación. **Contacto: jlsierra@ucm.es**