Proyecto Andrómeda - Software de astrología Armon2

Implementación de técnica astrológica de Direcciones Simbólicas

Por Héctor Jiménez – Febrero 2026

Introducción

Este documento tiene como fin explicar las definiciones, decisiones y pasos de implementación que se llevaron a cabo en el desarrollo en código de programación de computadora para la técnica de Direcciones Simbólicas usadas en astrología.

Historia

El propósito del proyecto Andrómeda es muy Neptuniano, me salen estas "visiones" de cómo integrar lo mejor de los mundos, conceptos y tecnología, en una sola "burbuja". Si veo atrás en los caminos que he tomado en mi vida, en mi interior surge el deseo de unir mis dos conocimientos que en realidad me apasionan: la astrología y las ciencias computacionales.

Trasfondo

La intención de este documento es que sirva de entendimiento sobre el proceso que se lleva a cabo para implementar en código de programación y que los usuarios de Armon2 sepan con mayor exactitud cómo y porqué ha sido implementada.

Este documento describe, con enfoque técnico, la implementación de Direcciones Simbólicas en Armon2.

Historia y conceptos

En astrología predictiva, una "dirección" es una **transformación** (mapeo) de la carta natal hacia un "estado dirigido" asociado a una edad o fecha, con el fin de comparar **puntos dirigidos** (planetas, ángulos, cúspides) vs. **puntos natales** (u otros puntos dirigidos) y obtener contactos (aspectos, conjunciones, etc.) con significado temporal. En el sentido más general:

- Se tiene una carta natal (radix) con longitudes eclípticas λ (y a veces latitudes β).

- Se define una **clave** (llave) que convierte tiempo a arco:

$$\text{arco} = K \cdot \text{edad}$$

- Con ese arco se "mueven" **longitudes eclípticas** (zodiacal/eclíptico), o bien, **ascensión recta** (AR, método ascensional), o bien, el **ARMC/RAMC** (para mover casas de forma coherente).

Existen variantes clásicas:

A) Direcciones por "arco fijo" (1 año por grado) - Forma simplificada: se asume que a cierta edad corresponde cierto arco, por ejemplo, **Ptolomeo**: $K = 1°/\text{año}$ o **Naibod**: $K \approx 0.98564733°/\text{año}$ ($\approx 360/365.2422$). Esto se usa como aproximación "simbólica" o "regular".

B) Direcciones "Solar Arc" (Arco Solar) - Otra familia: el arco se define por **el avance del Sol progresado**:

$$\text{arco} = \lambda_{\odot,\text{prog}} - \lambda_{\odot,\text{natal}}$$

y ese mismo arco se aplica a todos los puntos.

C) Direcciones primarias "Placidus / Zodiacal" (speculum + house position) - Aquí el enfoque no es "sumar un arco a la longitud", sino dirigir el **ARMC** con la clave, recalcular casas con ese ARMC dirigido, y para cada planeta/punto, conservar su **posición de casa** (house position) y resolver qué longitud eclíptica dirigida produce la misma posición de casa bajo el ARMC dirigido.

La implementación de Direcciones Simbólicas en Armon2 corresponde a la variante B descrita anteriormente. En el módulo implementado en Armon2, `astro_engine.py`, existe una función `primary_directions(...)` que implementa el enfoque "Solar Arc" (Sol progresado secundario para obtener el arco + recalcular casas por ARMC) descrito en este documento. Existe otra función en el código llamada `primary_directions_placidus_zodiacal(...)`, esta implementa las direcciones primarias "puras", no confundir con Direcciones Simbólicas ésta última.

Notas sobre cálculos

El programa usa las funciones de la librería Swiss Ephemeris para realizar varios cálculos importantes.

El parámtero de `ECL_NUT` que se menciona más adelante, es una constante de *Swiss Ephemeris* (la librería swisseph que se usa en *astro_engine.py*) que indica que se quiere calcular la **oblicuidad de la eclíptica** (ε) y la **nutación** para una fecha dada. La oblicuidad de la eclíptica es el ángulo entre el plano de la eclíptica (órbita aparente del Sol) y el ecuador celeste. Actualmente es aproximadamente:

$$\varepsilon \approx 23°26'$$

Pero cambia lentamente con el tiempo (≈ 0.013° por siglo).

La nutación es una oscilación pequeña y periódica en la orientación del eje de rotación de la Tierra. Si la precesión es el movimiento lento y continuo del eje terrestre (≈ 26,000 años por ciclo), la nutación es un pequeño "temblor" superpuesto a ese movimiento. La nutación se debe principalmente a la atracción gravitatoria de la Luna y en menor medida, la del Sol. La Luna es la responsable principal del término más grande de nutación. La nutación tiene una periodicidad principal, el término más importante de la nutación tiene un período de

$$18.6 \text{ años}$$

que corresponde al ciclo de los nodos lunares.

El programa necesita ε porque al conviertir Longitud eclíptica (λ) a Ascensión Recta (RA) la fórmula depende de ε. Se necesita esta conversión no para los cálculos de los planetas dirigidos sino para la casas, las casas Placidus no se calculan en longitud eclíptica, sino en el sistema ecuatorial (Ascensión Recta). Por eso se necesita convertir el MC dirigido (que se movió en longitud) a AR para poder recalcular correctamente las casas.

Las direccione simbólicas si mueven a los planetas sobre la eclíptica (longitud eclíptica λ), las casas Placidus no están definidas puramente en la eclíptica. Placidus es un sistema basado en la rotación de la Tierra, tiempo sidéreo, Ascensión Recta (AR) y meridiano local, es decir, las casas viven en el sistema **ecuatorial**, no en el eclíptico.

El MC no es simplemente "un grado de la eclíptica". Astronómicamente, el MC es el punto de la eclíptica que está cruzando el meridiano local. Pero el meridiano se define en coordenadas ecuatoriales (RA). Por eso, en *Swiss Ephemeris* `swe.houses_armc(armc, lat, eps, hsys)`. Lo que se necesita "pasar" es ARMC (Ascensión Recta del Medio Cielo), latitud geográfica y la oblicuidad (ε).

Geométricamente debemos pensar en 2 planos distintos:

Plano 1 — Eclíptica - Donde viven el Sol**,** planetas**,** arco solar**,** longitudes zodiacales

Plano 2 — Ecuador celeste - Donde viven Ascensión Recta**,** ARMC**,** cálculo de casas**,** rotación terrestre

La oblicuidad ε es el ángulo entre esos planos por lo tanto, sin ε no podemos pasar de uno al otro. Para Direcciones Simbólicas el cálculo básico que el código implementa es:

```
armc_dir = degrees(atan2(sin(lam)*cos(epsr), cos(lam))) % 360.0
```

Eso matemáticamente es:

$$\alpha = \text{atan2}\ (\sin\ \lambda\cos\ \varepsilon, \cos\lambda)$$

Sin ε, no se puede pasar correctamente de coordenadas eclípticas a ecuatoriales. La implementación usa entonces `ECL_NUT` en vez de una constante fija como 23.44° porque es astronómicamente correcto: La oblicuidad cambia con el tiempo. *Swiss Ephemeris* calcula la **oblicuidad verdadera** para el JD exacto y eso garantiza que el ARMC sea preciso, las casas Placidus sean precisas y la conversión RA sea coherente. Si el programa usara un valor fijo, habría pequeñas desviaciones en casas y ángulos (especialmente en validaciones de precisión). El código usa ECL_NUT porque implementa la **oblicuidad verdadera**, no solo la media (sería el valor de `ECL` de *Swiss Ephemeris*).

Direcciones Simbólicas en Armon2 - Definición operativa

En la implementacíon del programa, "Direcciones Simbólicas" se implementa como **Solar Arc (Arco Solar) en longitud eclíptica**, con:

- **Arco Solar** calculado a partir del **Sol progresado secundario** (método 1 día = 1 año).
- **Todos los planetas/puntos** se desplazan por el **mismo arco** en **longitud eclíptica**.

- **Casas/ángulos**: el programa usa el enfoque **"SA en Longitud" para el MC**, y después recalcula casas **Placidus** a partir del **ARMC** derivado del MC dirigido.

Esta técnica es "simbólica" porque aplica un **arco único y uniforme** a todo el mapa (planetas y estructura de casas), en lugar de usar semiarcos/RA clásicos de direcciones primarias.

Entradas y normalización temporal

Fechas - `primary_directions(birth_date, location, target_date, converse)`: `birth_date`: ISO (si viene con espacio lo convierte a T); `target_date`: si viene como solo fecha YYYY-MM-DD, se completa con la hora natal (si se puede leer de birth_date), si no 00:00:00.

Ubicación y zona horaria - Resuelve coordenadas y `tz` con `get_coords(..., location) → resolved_lat, resolved_lon, resolved_tz`**.** Luego convierte a tiempos (`to_time`) y pasa a UTC.

Edad y fecha progresada (secundarias 1 día = 1 año) - El programa define la edad como:

$$\text{edad}_{años} = \frac{(t_{target} - t_{birth})}{365.2422 \text{ días}}$$

(en segundos / (365.2422 × 86400)). Luego construye la **fecha progresada** en JD UT con la regla:

$$JD_{prog} = JD_{birth} + \text{edad}_{años}$$

Es decir: **suma "edad en años" como si fueran "días"** (1 día simbólico = 1 año real).

Cálculo del Arco Solar - la longitud del Sol Natal:

$$\lambda_{\odot,nat} = \text{swe.calc\_ut}(JD_{birth}, SUN)$$

y el Sol Progresado:

$$\lambda_{\odot,prog} = \text{swe.calc\_ut}(JD_{prog}, SUN)$$

luego el **arco solar directo** es:

$$A = (\lambda_{\odot,prog} - \lambda_{\odot,nat}) mod 360$$

y si el usuario en la interfaz gráfica marca **converso**:

$$A \leftarrow -A$$

```
arc = (sun_prog - sun_nat) % 360.0
if converse:
    arc = -arc
```

Aplicación del arco a planetas/puntos (zodiacal en longitud)

La carta natal base se obtiene con `natal = chart_for_date(birth_iso, lat, lon, tz, location)`, y luego:

$$\lambda_{dir}(p) = (\lambda_{nat}(p) + A) mod 360$$

cuya implementación literal es:

```
planets_raw_dir = {k: (float(v) + arc) % 360.0 for k, v in planets_raw_nat.items()}
```

NOTA: aquí "puntos" son los que existan en `planets_raw natal` (Sol, Luna, planetas, etc., según la función interna de `chart builder`). **Todos se mueven idéntico**.

Ángulos y casas: "SA en Longitud" para el MC + ARMC + Placidus - Esta parte es lo que diferencia la implementación a no sólo un "simple shift" de cúspides. Para el cálculo del MC dirigido en longitud, se toma el MC natal desde `houses_raw_nat["mc"]` y se aplica el arco:

$$MC_{dir,\lambda} = (MC_{nat,\lambda} + A) mod 360$$

el código que implementa esto es `mc_dir_long = (mc_nat + arc) % 360.0`

Oblicuidad – Se toma la oblicuidad (en la fecha natal):

$$\varepsilon = \text{swe.calc\_ut}(JD_{birth}, ECL\_NUT)$$

ARMC "correspondiente" al MC dirigido (conversión eclíptica → RA) – El programa convierte la longitud eclíptica del MC dirigido a ascensión recta (RA), usando:

$$ARMC_{dir} = \text{atan2}(\sin\lambda \ \ \cos\varepsilon, \cos\lambda)$$

(en grados, normalizado a 0–360), el código literal es

```
lam = radians(mc_dir_long)
epsr = radians(eps_deg)
armc_dir = degrees(atan2(sin(lam)*cos(epsr), cos(lam))) % 360.0
```

Esto implementa exactamente la idea "SA en Longitud del MC → RA equivalente → usar ese RA como ARMC para casas".

Recalcular casas Placidus con houses_armc - con ese `armc_dir` se re-calculan las casas con `cusps, ascmc = swe.houses_armc(armc_dir, resolved_lat, eps_deg, hsys)` y se guarda:

- asc = ascmc[0]
- mc = ascmc[1]
- cusps = list(cusps)

"Fallback", si falla houses_armc - Si por cualquier razón falla, el fallback implementado es un desplazamiento simple: cúspides = cúspides natales + A**;** asc = asc natal + A; y el mc = mc natal + A

Ejemplo

Como un ejemplo breve conceptual, si para cierta fecha:

- $\lambda_{\odot,nat} = 15.0°$
- $\lambda_{\odot,prog} = 44.5°$

Entonces:

$$A = 29.5°$$

- Marte natal $= 100.2°$ → Marte dirigido $= 129.7°$
- MC natal $= 210.0°$ → MC dirigido (longitud) $= 239.5°$
- conviertes $239.5°$ a RA con ε natal → armc_dir

con armc_dir calculas Placidus (ASC/MC/cúspides)

Pseudocódigo - función primary_directions(...)

```
function direcciones_simbolicas_solar_arc(birth_iso, location, target_iso, converse):
  (lat, lon, tz) = get_coords(location)
  t_birth  = to_time(birth_iso, tz)
  t_target = to_time(target_iso (si solo fecha -> usa hora natal), tz)
  jd_birth  = julday(t_birth_utc)
  jd_target = julday(t_target_utc)
  age_years = (t_target_utc - t_birth_utc) / (365.2422 días)
  jd_prog = jd_birth + age_years  # 1 día = 1 año
  sun_nat  = lon_sun(jd_birth)
  sun_prog = lon_sun(jd_prog)
  arc = (sun_prog - sun_nat) mod 360
  if converse: arc = -arc
  natal = chart_for_date(birth_iso, lat, lon, tz)
  # Planetas: todos el mismo arco
  for each point p in natal.planets_raw:
      p_dir = (p_nat + arc) mod 360
  # Casas:
  mc_nat = natal.houses_raw.mc
  mc_dir_long = (mc_nat + arc) mod 360
  eps = obliquity(jd_birth)
  armc_dir = RA_of_ecliptic_longitude(mc_dir_long, eps)
  try:
      casas_dir = houses_armc(armc_dir, lat, eps, Placidus)
  except:
      casas_dir = shift_simple(natal.cusps, arc)
  return chart_dir(planets_dir, casas_dir, meta: age_years, arc, ...)
```

Lo que el código no implementa

- **No** es una Dirección Primaria clásica por semiarcos.

- **No** es la función `primary_directions_placidus_zodiacal(...)` (esa es el módulo de **Direcciones Primarias**: zodiacales "puras" por house_pos/inversión, o AR-eclíptico según modo).
- En Direcciones Simbólicas, no se conserva `house position` ni se hace una inversión numérica por Placidus: aquí el movimiento planetario es **"+ arco" directo en longitud**.