

Time Harmonisation and Temporal Boundaries

Introduction

Time harmonisation across a heterogeneous system is a key component of the HELIXsystem enabling the observed system interactions to be sorted into their logical temporal order by the Assembler Utility.

Temporal harmonisation is not temporal synchronisation. The purpose of temporal harmonisation is to introduce bounded temporal tolerances to enable the digital representation of the executing environment to be constructed and maintained.

To deliver this common temporal reference the HELIXsystem uses a modified-standard approach based on the Precision Time Protocol (IEEE 1588). PTP prescribes a “Master-Servant” structure with the “Master Clock” located in the Assembler Utility and the “Servant Clocks” being located on the Collectors.

The Master Clock is programmed to send periodic updates to the Servant Clocks every n -seconds, providing continuously updating “coarse temporal alignment” across the distributed infrastructure.

The HELIXsystem modification is at the Collector level. As the agentic Collectors are able to communicate with each of their neighbours, this peer-to-peer communication enables clock drift and jitter to be minimised by the Collectors checking with their neighbours every y -seconds ($y < n$) for temporal alignment

This delivers temporally bounded system observability across a heterogeneous implementation enabling high-fidelity temporally bounded concurrent ordering to be established under controlled conditions.

Keywords: Heterogeneous systems, common temporal reference, Precision Time Protocol IEEE 1588, legacy implementations, distributed infrastructure, peer-to-peer communication across agents, temporally bounded system observability.

Interval Timestamps

Deterministic interaction ordering can be implemented using Interval Timestamps rather than a (T_i) point-in-time. This is a well recognised approach used across the industry, one example being Google’s “Spanner TrueTime”. This uses ($T_{\text{earliest}}, T_{\text{latest}}$) to define the temporal interval within which an event can be identified as having occurred.

One contingency that may arise when dealing with a heterogeneous network of Autonomous Agent’s (AA’s) implementing actions onto a shared state infrastructure, is two agents simultaneously implementing actions onto one communication construct, resulting in a race condition with the receiving endpoint unable to use (ΔT) to establish a deterministic ordering between the two actions. This can be expressed as:

$$\bar{x}(a) . P \mid \bar{x}(b) . Q \mid x(z) . R$$

This results in the receiving endpoint using a nondeterministic selection between the two competing reductions resulting in either:

$$\bar{x}(a) . P \mid x(z) . R \rightarrow P \mid R\{a/z\}$$

or

$$\bar{x}(b) . Q \mid x(z) . R \rightarrow Q \mid R\{b/z\}$$

As the selection is mutually exclusive, the non-selected action transitions from (S_0, T_0) to (S_1, T_1) . This invalidates the Verification Attribute (VA_i) associated with the AA's action, even though it has been independently determined. This can be addressed by introducing an "Interval Timestamp" defining the temporal parameters between which an action remains executable against the (VA_i) determination. As there is no realistic mechanism of securing time synchronisation across a heterogeneous environment introducing an "executability temporal buffer" aligns with temporal harmonisation.

Determining the Temporal Buffer

The determination of the Temporal Buffer is a function of the process requirements. This is best illustrated by example:

- AA's distributed across an algorithmic trading environment may need to execute their actions within millisecond bounded tolerances
- AA's distributed across a marine based supply chain system may only require bounded tolerances measured in days

The purpose of the buffer is to define a temporal tolerance interval relevant to the process being evaluated. This requires the event relationship tolerances to be User Configurable.

This configuration then becomes an ambient input into the AA and attaches to agent's optimal output ($OptR_i$) at the AA's output boundary.

The (VA_i) synchronised reduction:

$$[VA_i = (P_i \wedge T_{P,i} \wedge RO_i \mid S_0, T_0)]$$

Where:

$$P_i = Process$$

$$T_{P,i} = Toplogy$$

$$RO_i = RequiredOntology$$

remains an output of the Deterministic Control Plane and remains bound to the authoritative state (S_0, T_0) .

The Temporal Buffer now becomes a component of the execution condition. This becomes the tuple:

$$(OptR_i, VA_i, B_i)$$

This enables the function of each of components to be clearly delineated:

$OptR_i$ = Optimal Response determined by the AA

VA_i . = (*Admit, Deny*) executability verification of $OptR_i$ against (S_0, T_0)

B_i . = the parameters of the User Defined Temporal Buffer

The Verification Attribute addresses whether the agent's proposed response is executable against the current authoritative state of the process, while the Temporal Buffer establishes the temporal conditions used when competing actions arrive simultaneously at a common endpoint.