Validation for Mapping Control and SAS properties

Date: 30/01/2021
Researchers: Ricardo Caldas, Razan Ghzouli, Patrizio Pelliccione, Danny Weyns, Thorsten Berger

1 Modeling CT properties

Here, we discuss the process of modeling stability, settling time, overshoot, and steady-state error. Where we present a definition for each property, a graphical representation that supports the understanding of the property, and a matching with well-known specification patterns[1].

1.1 Stability.

Definition. *"A system is said to be stable if for any bounded input, the output is also bounded." (Hellerstein et al., 2004)*

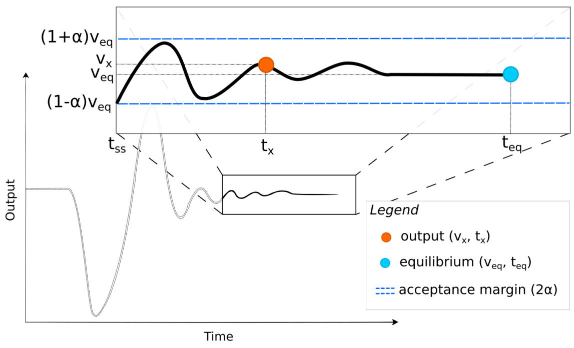

Figure 1: Graphical Representation of Stability

The Fig. 1 illustrates a set of output values (v_x) bounded by an acceptance margin. Where, the acceptance margin (α) is a limiting value relative to the equilibrium value (v_{eq}), which is the expected convergence value for the output. Thus, we define a stable state (S), where the difference between the system output value and the equilibrium value is bounded: $S \equiv |v_x - v_{eq}| < \alpha$.

A system is stable when it can always recover from unstable states (¬S). Thus, stability is modeled through the combination of the patterns Untimed Existence, which aims at describing a portion of a system's execution that contains an instance of certain events or states, and Untimed Universality, which aim at describing a portion of a system's execution that is free of certain events or states.

> After Untimed Existence & After Untimed Universality
>
> **After ¬S eventually it is always the case that S holds**

However, stability per se is a strong property and systems operating in uncertain environments might need bounded certification of stability. Hence, we formalize Weak Stability employing the pattern Globally Untimed Response, which aims at describing cause-effect relationships between a pair of events/states.

> Globally Untimed Response
>
> **Globally if ¬S , then in response eventually S holds**

1.2 Settling Time.

Definition. *"The settling time is the amount of time required for the signal to stay within $\alpha\%$ (acceptance margin) of its final value for all future times", (Åstrom A. et al., 2010)*

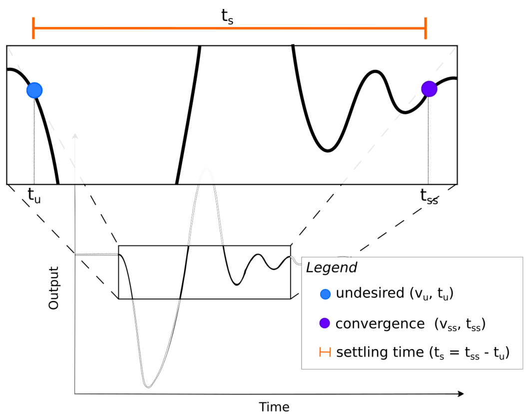

Figure 2: Settling Time graphical representation

Åstrom A. et al. define settling time as the amount of time (t_s) required for the output value to converge. We complement it by expressing t_s as accumulated time from the moment (t_u) when the output signal is at the undesired value (v_u) until the state S is reached. We employ the pattern Timed Response, which aims at describing cause-effect relationships between a pair of events/states within a timebound.

> Globally Timed Response
>
> **Globally if ¬S then in response S eventually holds within $t_s \in \mathbb{R}+$ time units**

[1] http://ps-patterns.wikidot.com/

1.3 Overshoot

Definition. *"The overshoot is the percentage of the final value by which the signal initially rises above the final value", (Åstrom A. et al.,2010).*

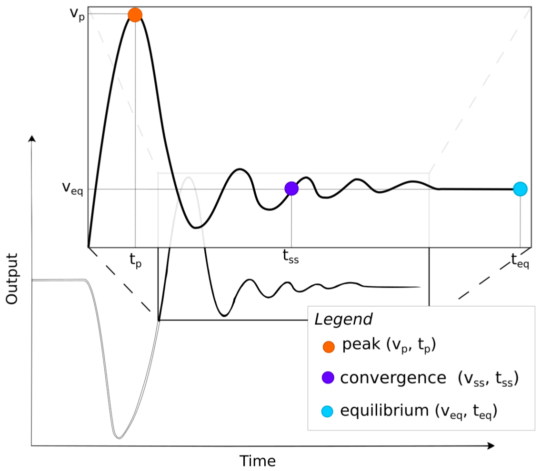

Figure 3: Overshoot graphical representation

The Fig. 3 illustrates that the overshoot is a state (OS) in which there is an upper limit (v_{ul}) to the relative difference between the peak value (v_p) and the equilibrium value (v_{eq}): $OS \equiv \frac{v_p - v_{eq}}{v_{eq}} 100\% < v_{ul}$. We model overshoot employing the pattern Untimed Absence, which aims at describing a portion of a system's execution that is free of certain events or states.

Before Untimed Absence

Before S**, it is never the case that** ¬OS.

1.4 Steady-State Error

Definition. *"If the output of a system at steady state does not exactly agree with the input, the system is said to have steady-state error. This error is indicative of the accuracy of the system", (Ogata K., 2002)*

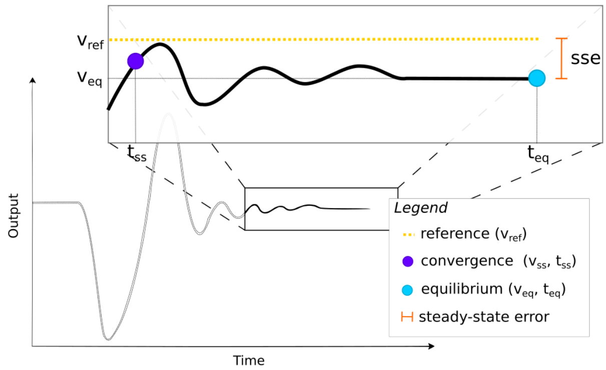

Figure 4: Steady-State Error graphical representation

The Fig. 4 depicts the steady-state error as the difference between the reference value (v_{ref}), and the the equilibrium value (v_{eq}). This property holds in a state (SSE) under a limiting value (v_l) that bounds the distance between reference and equilibrium: $SSE \equiv |v_{ref} - v_{eq}| < v_l$. Therefore, we model steady-state error using the pattern Untimed Absence.

After Untimed Absence

After S**, it is never the case that** ¬SSE.

2 Mapping CT and SE properties

Table 1: Mapping between CT properties and SE properties.

ID	SE Requirement	SE Property	Pattern	CT Property
P1	Functional	Liveness	Globally Untimed Response	Weak Stability
P2	Functional	(2x) Reachability	Globally Timed Response	Settling Time

In P1, Tahara et al. validate their proposed verification tool (CAMPer) using the Tele-Assistance System (TAS) as a running example and elicit a property to guarantee that TAS will eventually become "b2" once it becomes "b1": $\Box(b1 \implies \Diamond b2)$. This property enforces a causality relationship between "b1" and "b2" in all execution traces. Where "b2" represents the current state of the system evolution and "b1" is a *before evolution* state. This formulation cannot be seen as stability since the property does not guarantee that b2 will be reached.

In P2, Moreno et al. evaluate their approach for accelerating complex self-adaptation decisions by proposing two time-bounded reachability predicates that model the adaptation process. To guarantee reachability, the delayed reachability indicates that if the system is in configuration c, with the passage of one decision interval τ it will reach configuration c′. Where configuration c′ is the system state after adaptation (convergence state (S)) and configuration c is the system undesired state before adaptation process (unstable state (¬S)). This property resembles the settling time by bounding the adaptation period within τ. The immediate reachability is a special case of the delayed reachability with τ close to zero.

References

[P1]. Tahara, Yasuyuki, Akihiko Ohsuga, and Shinichi Honiden. ”Formal verification of dynamic evolution processes of UML models using aspects.” 2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). IEEE, 2017.

[P2]. Moreno, Gabriel A., et al. ”Decision-making with cross-entropy for self-adaptation.” 2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS). IEEE, 2017.