Modeling and Requirements on the Physical Side of Cyber-Physical Systems

Mats P.E. Heimdahl, Lian Duan, Anitha Murugesan, Sanjai Rayadurgam

Department of Computer Science and Engineering
University of Minnesota
200 Union Street, Minneapolis, Minnesota 55455
Email: {heimdahl, lduan, anitha, rsanjai}@cs.umn.edu

***Abstract*—In a cyber-physical system (a system where the physical world interacts extensively with—often networked—software), the physical portion of the system resides in the continuous and continual domain. Thus, on the physical side of cyber-physical systems we will have to contend with not only real time requirements but also the continuous and continual nature of the system.**

This poses a new set of challenges for requirements engineering; we must write well defined requirements to address crucial issues not commonly addressed in the software domain. For example, the rate of change of a controlled variable, the time it takes for a controlled variable to settle sufficiently close to a set-point, and the cumulative errors built up over time may be of critical importance. In this paper we outline how *early modeling in the continuous domain* serves as a crucial aid in the elicitation and discovery of requirements for cyber-physical systems and provide an initial classification of the types of requirements needed to describe crucial aspects of the physical side of a cyber-physical system.

***Index Terms*—Requirements, modeling, cyber-physical systems;**

I. Introduction

Systems where the physical world interacts extensively with—often distributed and networked—software are today referred to as Cyber-Physical Systems (CPS). In such systems, the physical portion always resides in the *continuous* and *continual* domain; phenomena of interest are continuous, they represent quantities changing smoothly without discrete steps, and continual, the changes in the phenomena continues uninterrupted over time. Thus, providing requirements that discuss the ordering of events (as is done with various temporal logics) and the real-time properties of a system are not sufficient.

The continuous and continual aspects pose a new set of challenges for requirements engineering; we must write well defined requirements to address crucial issues not commonly addressed in the software domain. For example, the rate of change of a physical phenomenon controlled by the system (often called a controlled variable [1]), the time it takes for a controlled variable to settle sufficiently close to the desired value (the set-point), and the cumulative errors built up over time may be of critical importance.

This work has been partially supported by NSF grants CNS-0931931 and CNS-1035715.

In our work, we have been faced with several cyber-physical systems where the requirements necessary to adequately circumscribe the allowable systems behaviors have been quite difficult to elicit and define. Consider two on surface similar CPS control problems that turn out to necessitate quite different requirements; maintaining a car's speed through cruise control and maintaining the flow of a drug in an infusion pump. For instance, in the cruise control, requirements on the how quickly the speed is brought up to a new set-point and how quickly the speed has to stabilize around that set-point are crucial requirements for passenger comfort, reduced wear on the drive-train, and fuel economy [2]. In the infusion pump, on the other hand, the system dynamics make such requirements superfluous and irrelevant, bringing the flow-rate of the drug up a higher level can be done quickly without harm to the patient and the flow will stabilize around a new set-point quickly since there is little inertia in the system. As another example, the cumulative error in the speed of the vehicle in a cruise control is manifested in the distance we have travelled at a point in time; a measure of no relevance to the driver. In an infusion pump, the cumulative error in flow-rate is manifested in the volume of drug that has been infused in the patient; a measure that can have a life or death impact on the patient.

To discover, define, and, in some cases, justify exclusion of CPS requirements, *early modeling in the continuous domain* has served as a crucial aid. In effect, these models are proposed solutions to the problem at hand; they are early prototypes of a small aspect of the proposed system. For example, we may build a simplified model of the vehicle dynamics and the controller for a cruise control system. This model resides in the solution domain—it is a proposed behavior for our new system—since it describes in detail how to change acceleration based on deviations in actual vehicle speed from the desired vehicle speed. The cruise control requirements, on the other hand, reside in the problem domain and may define aspects such as the accuracy of the cruise control (how close to the desired speed should the actual speed be) and time allowed to bring the vehicle up to the desired speed when resuming cruise control after braking (the cruise control is disengaged when braking). The requirements elicitation and definition form a symbiotic relationship with the behavioral modeling, a relationship where initial requirements constrain the modeling efforts and model behavior leads to requirements discovery—a desirable cycle similar to what Nuseibeh described in his work on Twin Peaks [3].

In this paper we briefly discuss our modeling efforts and provide an initial classification of the types of requirements needed to describe crucial aspects of the physical side of a cyber-physical system—requirements elicited and discovered through the modeling efforts.

II. Example Systems

In the paper we will use two example systems, a basic automotive cruise control requiring little explanation and a drug infusion pump discussed a bit more extensively below.

Our basic automotive cruise control is a control system whose purpose is to maintain the vehicle's speed at a predefined speed. The system is disengaged when the driver uses the brakes or the throttle, and reengaged when the driver requests the cruise control to resume.

Infusion pumps are medical cyber physical systems used for controlled delivery of liquid drugs into a patient's body according to a physician's prescription, a set of instructions that governs the plan of care for that individual.Patient-Controlled Analgesia (PCA) is type of infusion pump that is generally equipped with a feature that allows patients to self-administer a controlled amount of drug, typically a pain medication.

Infusion pumps generally provide multiple modes of drug delivery. In *basal* mode, the drug is delivered at a constant (and usually low) rate for an extended period of time. In a *bolus* mode, the drug is delivered at a higher rate for a short duration of time to address some immediate need. There may be multiple bolus modes. In *clinician bolus* mode, the drug is delivered at an elevated rate in response to a clinician's request. Further, in a PCA system, a *patient bolus* mode may be activated to deliver additional drug in response to a patient's request for more medication, typically to alleviate acute pain.

III. The Twin-Peaks of Models and Requirements

In our work we have found that identifying all the requirements that must be stated to address the complexity in the continuous domain has been a challenge. We have found little guidance in the requirements engineering literature as well as in the control systems community, who focus on the development of control algorithms but not how to state the constraints (the requirements) that circumscribe the control solution.

Consider, for example, the requirements below paraphrased from the automotive cruise control and infusion pump domains.

> *"When the driver requests the cruise control to resume, the cruise control shall be engaged and bring the vehicle's speed to the target speed."*

> *"A patient bolus dose shall be given when requested by the patient."*

Although these two requirements are from two radically different domains, the nature of the requirements is identical; they are both examples of what the system must do when there is an abrupt change in the set-point (the desired speed or desired flow-rate). In the cruise control example, the vehicle might be coasting at 80 km/h with no desired speed (the cruise control is inhibited) and the resume request abruptly sets the desired speed to 110 km/h. In the infusion pump, the desired basal flow-rate might be 5 ml/h and the bolus request abruptly sets the desired flow-rate to the patient bolus value of 7 ml/h.

Such requirements may be accompanied by accuracy requirements defining how close to the desired values (speed and flow-rate) the actual speed and flow-rate must be.

> *"The actual speed of the vehicle must be within $\pm 5\%$ of the target speed."*

> *"The actual flow-rate must be within $\pm 5\%$ of the target flow-rate."*

When looking at such requirements there were immediate questions regarding issues related to the changes in set-points. For example:
How much time are we allowed when trying to to bring the actual speed (or flow-rate) to the desired speed (or flow-rate)? and
How fast are we allowed to change the speed (or flow-rate)? Requirements covering these aspects of the systems were not captured in the documentation we reviewed. To investigate such questions and gain a better understanding of the requirements we resorted to modeling of the physical aspects of the system.

The models we developed are traditional control models including the controller, sensor and actuator models, as well as a plant model. Figure 1 shows an example of such models (in this case for the GPCA infusion pump). These models are traditionally used to evaluate various control strategies, tune the controllers, etc. In our case, however, we developed the models to better understand the requirements needed to adequately constrain the desired system behavior.

Through these modeling efforts one has an opportunity to explore various system responses and investigate how a proposed system might behave in its intended environment. The scenarios mentioned above (sudden change in the desired speed or flow-rate) are often used as a baseline evaluation of the response of a proposed control approach—typically referred to as the step response in the controls literature [4]. An example of the step response for the cruise control system is shown in Figure 2.

A. Learning From the Step Response

The step response in Figure 2 was generated from a Simulink [5] model of the simple cruise control. At time $t = 5$ seconds, the input step function goes from 80 km/h to 110 km/h (shown in red dotted line). The response is shown with the blue solid line. When looking at this simple graph there were several aspects of this system that caught our attention,

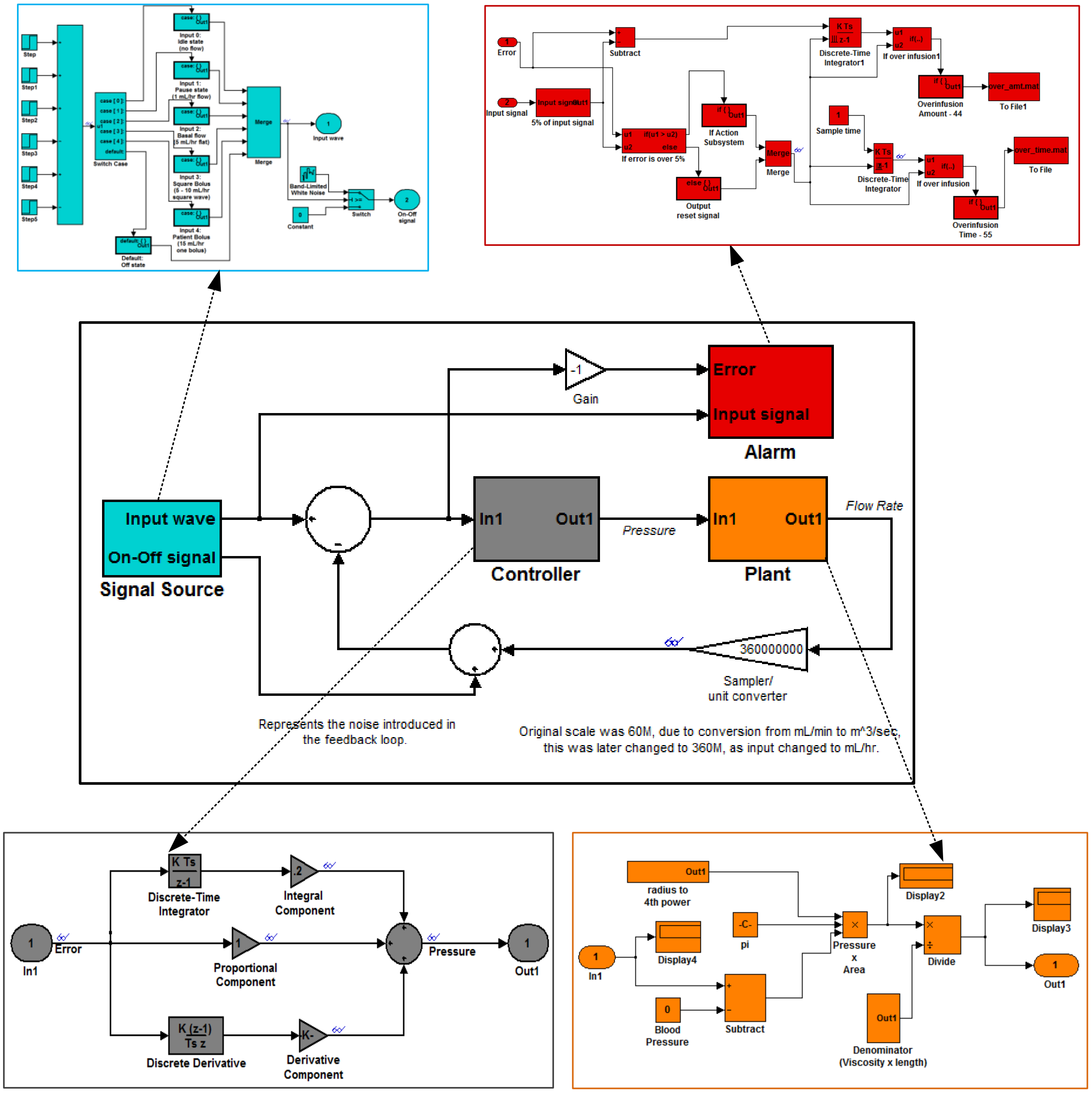

Fig. 1. Control System Model

for example, how quickly is the controlled variable allowed change, how much time is allowed before it reaches its desired value, and how much is it allowed to initially overshoot the desired value.

In the controls literature [4], many of the attributes of the step response are discussed. The first variable to be considered is the rise time. The rise time is defined in [4] as the time it takes for the step response to increase from 0% to 100% of its expected final value. How fast this rise-time is can have important ramifications on the system depending upon the application domain. For instance, in the case of an infusion pump, a rise time that is too fast is largely irrelevant (in fact, from discussions with our industry collaborators, for some pumps it is assumed that there is an instantaneous rise time), while a rise time that is too slow could potentially delay the treatment of a patient. On the other hand, for a vehicle's cruise

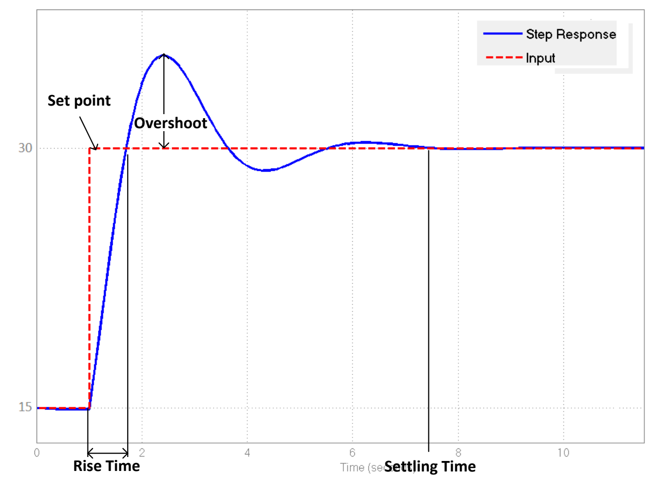

Fig. 2. The step response for the automotive cruise control.

control, a sudden and fast change in speed could be extremely uncomfortable to the passengers in the vehicle.

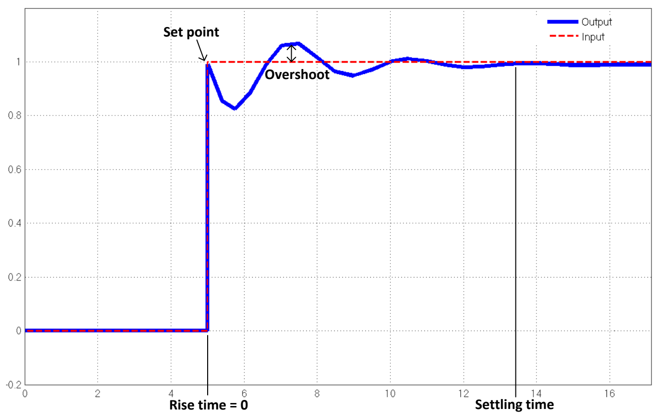

Fig. 3. The step response for the GPCA.

Another aspect of the graph that could be of interest is the overshoot. This is how much beyond the maximum desired final value (110 km/h, in the above example) the response reaches. In the case of an infusion pump, often times a maximum flow-rate is typically explicitly specified in the requirements. The overshoot can violate this value and a model can give extra insight into where such situations may occur.

The settling time is the time it takes the step response to reach its final value (the set point). Oscillations during the settling time can cause passenger discomfort (car sickness) in a cruise control system or highly inaccurate infusion over short time periods in an infusion pump.

The set point is the desired final value of the system response. In all systems there is a range above and below the set point that is still considered a "valid" response; due to the variabilities (noise) in the real world the control cannot be 100% accurate, we must allow for some deviation from the desired value. Current requirements specifications typically specify such a tolerance, but such tolerances must be treated carefully and can potentially bring up issues related to accumulated error that might need to be considered in the requirements (see discussion below).

B. Accumulating Error

When the output is consistently within the acceptable limits, but is also consistently above the set point, as seen in Figure 4, some issues may emerge. Based on the requirements for the GPCA pump, it has been assumed that the output flow rate is the desired flow rate. Nevertheless, due to limitations in the physical world, this is generally not possible. Such fluctuations can be modeled with the addition of a noise component in the sensors, actuator, and/or plant models. If this noise contains a constant positive offset, the disturbance may be enough to increase the actual flow rate by a small amount while still remaining within the acceptable tolerance. As such, there is a possibility of a patient may receiving too much of a drug over a period of time, the error accumulates and may build up to a real problem. For instance, suppose a doctor has prescribed a patient to receive an infusion of 5 ml/hr. The acceptable tolerance for the infusion pump is 5%, allowing infusion amounts to vary between 4.75 and 5.25 ml/hr. If the machine infused at the 5.25 ml/hr rate for an entire day, then at the end of a 24 hour period the patient will have received an extra 6 ml of the drug as compared to the expected volume—the equivalent of more than an extra hour of infusion.

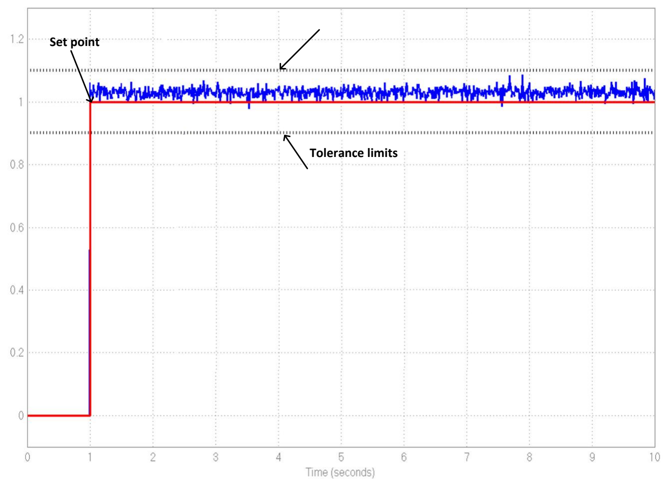

Fig. 4. The error between the target flow-rate and the actual flow-rate in the GPCA infusion pump.

C. Discussion

The modeling of candidate control approaches for various aspect of a system, such as the speed hold feature of a cruise control system or the infusion modes of a GPCA infusion pump provides insight into the requirements, in particular into the requirements governing the various aspects of the continuous phenomena in a cyber-physical system. As requirements are discovered or refined, the models are brought into conformance with new requirements, and new requirements will be discovered or refined. As an example, an initial control approach for the cruise control may be very slow and gradually bring the vehicle up to the desired speed. At a review, this sluggish response is deemed unacceptable and requirements addressing the rise-time are added and the control approach changed to implement this shorter rise-time. This quicker response may now lead to a large overshoot of the target speed and oscillation, something that was not an issue with the more gradual speed increase in the previous version. Requirements to address these new problems must now be added and old requirements might have to be modified; it may be impossible to achieve the required rise-time without a too large overshoot. The requirements and the models serve as our Twin Peaks assisting in finding and refining requirements in the problem domain by exploring candidate behaviors in the solution domain.

Naturally, during modeling we may discover that certain phenomena are nor relevant in the system under development. For example, in our GPCA infusion pump, the system dynamics allow for a very rapid rise time (there is no inertia to speak of in the system) and overshoot and settling time were deemed to be negligible for the same reason. Thus, one can argue that requirements governing these quantities are superfluous for the GPCA pump. Without modeling in the continuous domain, discovering which requirements are needed and which ones are superfluous (and then documenting the reasons why they are deemed superfluous) will, in our opinion, simply not be possible.

IV. Initial CPS Requirements Classification

As mentioned above, during our iterative requirements and modeling efforts working on the GPCA infusion pump, we realized that classes of requirements related to the continuous and continual behavior of the system were not captured in the available documentation. These missing or inadequate requirement generally fell into six categories closely related to the properties of the step response curves discussed in the previous section. The classification presented in this section is intended to be a first step towards a catalog of *cyber-physical systems requirements* that can serve a guide or checklist for developers of such systems; these classes of requirements should be present in a requirements document or their absences should be justified.

To illustrate the requirements we will be primarily using the Actual Flow Rate of drug through the infusion hose in an infusion pumps system. Note that the requirements examples presented in this section are simply examples of what such requirements might look like; it is not intended to be a catalogue of CPS requirements patterns. There are numerous ways of specifying the various quantities in question and we hope to provide a more comprehensive statement in forthcoming work.

Note that the requirements examples presented in this section are simply examples of what such requirements might look like; it is not intended to be a catalog of CPS requirements patterns. There are numerous ways of specifying the various quantities in question and we hope to provide a more comprehensive account in forthcoming work.

A. Accuracy

The accuracy (or precision) of the control of a physical quantity is quite obviously a concern and is typically defined in a requirements document. In our case, the accuracy of the flow-rate can be defined as a percentage of the flow-rate.

> *"The actual flow-rate (f) during normal operation shall be within ±5% of the target flow-rate (tfr): 0.95· tfr $\leq$ f $\leq$ 1.05· tfr."*

Naturally, the accuracy can be specified as an absolute value or as a value that decreases (as opposed to increases) with the flow-rate (the pump may be more accurate at higher flow-rates). Nevertheless, accuracy requirements for all controlled quantities are needed.

B. Rise Time (Drop Time)

The time allowed to go from one set-point—in our case typically basal flow-rate or a very low flow rate, such as Keep Vein Open (KVO) flow rate, to one of the bolus flow rates—cannot be more than a certain time interval. If the rise time is unconstrained, we may pick a solution that increases (or decreases) the quantity so slowly that it is harmful to the patient. In our case:

> *"The duration between the time at which a new target flow-rate (tfr) is commanded and the time at which the actual flow-rate (f) reaches within ±5% of the target flow rate shall be shall be less than 1.0 s."*

Requirements governing at least the maximum rise time (or drop time) are needed in all systems controlling physical quantities. A minimum rise time may be needed in case too rapid changes in a controlled quantity may be harmful. Such constraints on rise time could instead, however, be handled through requirements on the rate of change as opposed to rise time.

C. Rate of Change

Given a required rise-time, one can select different control approaches to reach the new set-point within the allocated time. For example, one can initially increase the flow rate rapidly and then taper off to smoothly capture the new target rate. Alternatively, one could rapidly increase the rate all the way to the new target rate and accept a larger overshoot (discussed next). Thus, in some instances (our automotive cruise control is a prime example), requirement on the rate of change may be needed—we may need to write requirement related to the first derivative of the controlled quantity.

> *"The rate of change in the actual flow-rate (f) shall not exceed 0.5 ml/s^2: $\dot{f} \leq 0.5\ ml/s^2$."*

In some systems we may even be interested in how quickly the rate of change changes. For instance, in our cruise control system, there will be requirements governing the rate of change in the speed (the acceleration of the vehicle—the first derivative of the controlled variable vehicle speed). In addition, we are most likely also interested in how quickly the acceleration changes (a concept known as *jerk*, the second derivative of the vehicle speed). Jerk is a crucial aspect of passenger comfort and constraints will need to be put on the acceptable jerk (think about how the negative acceleration when you brake for a red light suddenly goes away when the vehicle stops—you feel the "jerk").

D. Overshoot (Maximum Deviation)

The accuracy requirements on a system may be limited to its operation during normal operating conditions. During failure conditions, alarm conditions, and during transitions from one operational regime to another (often referred to as mode-changes), we may be willing to accept a larger deviation from the desired value—there may be an allowed maximum deviation (or maximum absolute quantity) that cannot be violated under any circumstance.

"The actual flow-rate (f) shall never exceed 10% of the target flow-rate (tfr): $f \leq$ *tfr +10%."*

"The actual flow-rate (f) shall never exceed 10 ml/h: $f \leq$ *10 ml/h."*

Requirements in this category will both limit the allowable overshoot as well as put safe limits on the various physical quantities.

E. Settling Time

In the tradeoff between rise time and overshoot, the system may experience some oscillation before the controlled quantities settle within the acceptable accuracy range. In our case, settling time turned out to be a non-issue (see Figure 3) since the system dynamics of the pump and fluid flow are not conducive to oscillation. If we want to constrain settling time, we need to fist define what "settling" means. In our case, we define selling as being within our normal operating accuracy, that is, within $\pm5\%$ of the target flow-rate.

"The time between when a new target flow-rate (tfr) is commanded and the time the actual flow rate (f) settles shall be less than 1.2 s."

Determining whether or not requirements on settling time are needed must be determined through modeling or previous system experience. If these requirements are omitted, a clear justification for their omissions should be presented.

F. Cumulative Error

As discussed in Section III, it is quite possible that there is a constant offset in the deviation between the actual values of a controlled variable and the desired values. Thus, errors may accumulate over time even though the system looks well behaved at any instance in time. In the case of the GPCA infusion pump, the volume infused in the patient might—due to infusion inaccuracy—become unsafe. Therefore, requirements on the cumulative error (in our case, the volume infused into the patient over a specified period of time) may be needed. If we assume that the time interval of interest in the GPC infusion system is δ second, we could constrain the volume infused as follows:

"The actual volume infused over a time interval of δ *cannot exceed the commanded volume to be infused by more than 0.1 ml:*
$\int_{t}^{t+\delta} f\,dt \leq \int_{t}^{t+\delta} tfr\,dt + 0.1$ *ml."*

Note here that if we want to express this requirement formally over the controlled quantities, we will need to express it as an integral over the flow rate since we are not directly controlling the volume of drug infused. In our work with the GPCA infusion pump, there have been a number of—often conflicting and highly unclear—requirements putting limits on the flow rate as well as the volume to be infused.

We advocate to identify the monitored and controlled variables at the interface between the environment, we are attempting to control and the system, we are putting in place to do so and express the system requirements in terms of this interface [6], [1]. Note that discovering this interface is a another "Twin Peaks" activity related to the architecture of the system solution we put in place and has been discussed elsewhere [7], [8], [9], [10].

V. Conclusion

In our work with various cyber-physical systems we have found that early modeling of the physical aspects of the proposed system is an invaluable resource when eliciting and clarifying requirements. The initial requirements constrain the initial modeling effort, the modeling effort raises questions and clarify thinking leading to new and better requirements, and the process is repeated.

We believe that the concerns of the control engineering community—concerns such as rise time, overshoot, and settling time—must be brought into the requirements engineering community to adequately constrain system behaviors and rigorously define what is considered an acceptable system behavior. Today, the details of the system's control behavior are often left to the control engineers using their expert judgement to determine if the control behavior is acceptable. We advocate codifying this expert judgement in the system requirements to ease, for example, system maintenance and evolution.

One issue not addressed in this report is the allowed system behavior when the system switches between different control approaches or the set-point changes (these changes are often associated with what we call mode-changes). Consider, for example, the simple change when the GPCA pumps switches from basal infusion to patient bolus infusion. At the moment we change the target flow rate from basal-flow to bolus-flow, the actual flow-rate will be severely out of tolerance (the system has not yet had time to bring the actual flow-rate up to the new target flow-rate). Thus, one may need a separate set of requirements defining the acceptable behavior during the various mode-changes. We have not yet investigated the mode-change issues in detail, but we hope to address these issues in our future work.

References

[1] D. Parnas and J. Madey, "Functional documentation for computer systems engineering," *Science of Computer Programming*, vol. 25, no. 1, pp. 41–61, 1991.

[2] Q. Xu, K. Hedrick, R. Sengupta, and J. VanderWerf, "Effects of vehicle-vehicle/roadside-vehicle communication on adaptive cruise controlled highway systems," in *Proceedings of Vehicular Technology Conference*, vol. 2. IEEE, 2002, pp. 1249–1253.

[3] B. Nuseibeh, "Weaving together requirements and architectures," *Computer*, vol. 34, pp. 115–117, 2001.

[4] R. Dorf and R. Bishop, *Modern Control Systems*. Pearson Prentice Hall, 2005.

[5] Simulink - simulation and model-based design. [Online]. Available: http://www.mathworks.com/products/simulink/

[6] M. Jackson, "The world and the machine," in *17th International Conference on Software Engineering*. IEEE, 1995, pp. 283–283.

[7] J. G. Hall, M. Jackson, R. C. Laney, B. Nuseibeh, and L. Rapanotti, "Relating software requirements and architectures using problem frames," in *Proceedings of Joint International Conference on Requirements Engineering*. IEEE, 2002, pp. 137–144.

[8] M. Brandozzi and D. E. Perry, "Transforming goal oriented requirement specifications into architecture prescriptions," in *Workshop from Soft. Req. to Arch*, 2001, pp. 54–61.

[9] P. Grunbacher, A. Egyed, and N. Medvidovic, "Reconciling software requirements and architectures: The cbsp approach," in *Proceedings of 5th International Symposium on Requirements Engineering*. IEEE, 2001, pp. 202–211.

[10] C. A. Gunter, E. L. Gunter, M. Jackson, and P. Zave, "A reference model for requirements and specifications," *IEEE Software*, vol. 17, no. 3, pp. 37–43, May/June 2000.