FORMAL SPECIFICATIONS OF SEQUENTIAL CONTROL FOR TRAINING MACHINES FOR THE LOWER LIMBS

J. Zaytoon[*], S. Moughamir[*], N. Manamanni[*], L. Afilal[*] and L. Angelloz[**]
[*]Laboratoire d'Automatique et de Microélectronique, University of Reims, France.
[**]Myosoft company. Bellegarde, France.

Abstract-**This paper presents a generic formal model for the specification and design of the sequential control for lower-limbs training machines. An object extension of Statecharts is used as a modeling formalism. The resulting hybrid and hierarchical control system interprets the required training specifications for a particular user to execute the corresponding sequence of switching (position, speed and force) control laws.**
Keywords **- Training and rehabilitation machines, hierarchical control, Statecharts, object design, Switching Control laws.**

I. INTRODUCTION

Training and rehabilitation machine for the lower limbs are becoming widely used today [1]. The design and the development of the control system for such machines is faced with a number of scientific and technical challenges. The domain-specific knowledge should therefore be exploited to provide assistance for this design and development process, by proposing a generic reusable model that can easily be specialised for the development of the control system for any particular training machine.

The main goal of the control system of a lower-limbs training machine is to execute a co-ordinated training sequence, alternating an upward movement and a downward movement. The type of the movement (velocity, position, and/or force controlled) is to be determined as a function of the required training mode for a particular user. The control system is also required to handle other tasks such as: initialisation, calibration of the effort measurement to take the gravity force into account, management of user's safety, as well as other specific features.

The aim of this paper is to propose a hierarchical formal model that capture the generic sequential control requirements related to the co-ordination and the execution of the identified movements and tasks. Section 2 presents the requirement analysis of the sequential control and co-ordination system and introduces an object extension of Statecharts, which is next used in Section 3 to specify the required generic sequential control model.

II. REQUIREMENT ANALYSIS AND SPECIFICATION FORMALISM

Training machines should be able to carry out different training configurations to comply with the diversity of requirements in medicine and sports. A preliminary analysis of different training configurations [2], [3] has shown that a training phase, in general, is characterised in terms of the *muscular group*, the *training mode*, the *training type*, and the *movement direction* involved (Table 1). These terms define the direction of the muscular force, the required movement type for a specific training or rehabilitation phase, the nature of the required muscular force, and the movement direction, respectively. Table 1 also shows that three control laws (position, velocity, and force) are required to carry out the different movement types.

TABLE I
CORRESPONDENCE BETWEEN MEDICAL AND TECHNICAL TERMS

Medical term	Corresponding definition
Muscular group	Direction of muscular force
- quadriceps	upward
- hamstrings	downward
Training mode	Movement type
- isokinetic	constant velocity with constraints on the muscular effort
- steering	constant velocity without constraints on the muscular effort
- isotonic	simulation of a constant heavy load
- physiokinetic	simulation of a heavy load as a function of the position
- assisted	velocity is a function of the muscular reaction, motor assistin this movement against gravit
- stretching	simulation of a heavy load, resulting in a low-speed movement to attain an equilibrium with the muscular effort
- isometric	limb maintained immobile while the user produces a muscular effort
Training type	Nature of muscular force
- concentric	muscle driving
- eccentric	muscle resisting
Limb movement	Movement direction
- extension	up
- flexion	down

This analysis also shows that a sequential controller for a training machine should be structured into three levels (Fig. 1). A training session comprises a number of successive phases. A phase corresponds to the execution of a number of series, each of which involves a specified number of repetitions of two movement patterns: an upward movement corresponding to an extension of the limb and a downward movement. The required movement pattern is selected as a function of the muscular group, the training mode and the training type characterising the current phase. Each movement pattern involves the execution of a corresponding sequence of position, velocity and force control laws, and the setting of the required reference value for the current control law. These switching control laws correspond to the continuous part of the control system, and should be specifically designed to cope with the mechanical structure and design of the target training machine. Two types of information flow can also be distinguished in Fig. 1: information exchanged with the machine (control signals and sensory information) and information exchanged with the user (orders and messages).

In order to manage the complexity, and to ensure the quality of the resulting sequential controller, the specification framework to be used should be based on a graphical formalism, easy to use by the domain expert, and expressive enough to capture the different aspects related to the co-ordination and the execution logic of the control tasks. This formalism should support hierarchical decomposition to model the 3-level sequential control structure depicted in Fig. 1. To satisfy these criteria, an object-oriented extension of Statecharts was used for the specification of the sequential

Report Documentation Page

Report Date	Report Type	Dates Covered (from... to)
25 Oct 2001	N/A	-

Title and Subtitle Formal Specifications of Sequential Control for Training Machines for the Lower Limbs	**Contract Number**
	Grant Number
	Program Element Number
Author(s)	**Project Number**
	Task Number
	Work Unit Number
Performing Organization Name(s) and Address(es) Laboratoire d'Automatique et de Microelectronique University of Reims, France	**Performing Organization Report Number**
Sponsoring/Monitoring Agency Name(s) and Address(es) US Army Research, Development & Standardization Group (UK) PSC 802 Box 15 FPO AE 09499-1500	**Sponsor/Monitor's Acronym(s)**
	Sponsor/Monitor's Report Number(s)

Distribution/Availability Statement
Approved for public release, distribution unlimited

Supplementary Notes
Papers from 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, October 25-28, 2001, held in Istanbul, Turkey. See also ADM001351 for entire conference on cd-rom.

Abstract

Subject Terms

Report Classification unclassified	**Classification of this page** unclassified
Classification of Abstract unclassified	**Limitation of Abstract** UU

Number of Pages
4

controller. This extension is based on the use of communicating modules related by hierarchical invocations. These notions are briefly presented below, and a more detailed presentation of this extended Statecharts formalism can be found in [4].

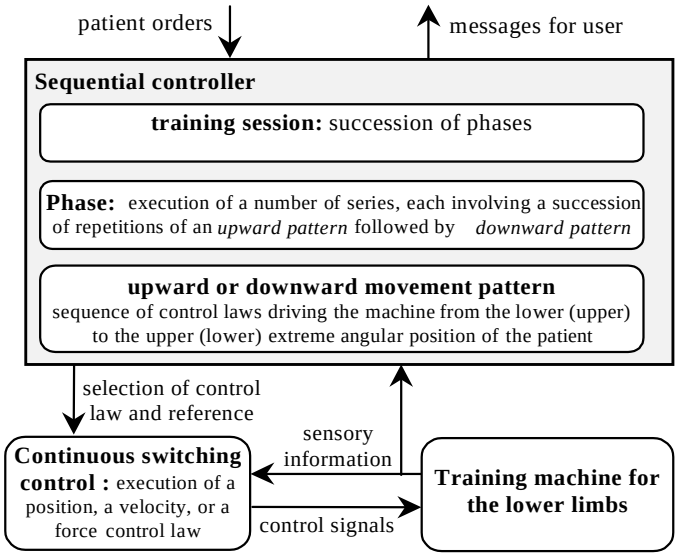


Fig. 1. Hierarchical control structure

A. Modules

With respect to a classical object structure, a module (Fig. 2) represents a class whose attributes are enriched with a dynamic behaviour that is common to all of the instances. This dynamic behaviour is specified by Statecharts [5]. Objects are defined as instances of modules.

A Statechart associated to a module characterises the behaviour common to its instances. This Statechart defines the logic of execution of the different tasks and provides a coherent and synthetic vision of the control flow. Statecharts, indeed, are a compact and expressive visual modelling formalism that extends conventional state-transition diagrams with the notions of hierarchy, concurrency a communication [5]. Activities can be executed in a given state using the primitive *throughout*. Operations can also be associated to the transitions of Statecharts. A transition is, therefore, labelled with the notation 'condition / operation'.

Module
Attributes attribute_1 : type attribute_2 : type ..... *Behaviour:* specified by Statecharts

Fig. 2. The module

B. Hierarchical invocation

Objects can be related hierarchically by means of the hierarchical invocation primitive "Invoke(object)". The use of this primitive within an "invoking" state of Statecharts results in the establishment of a hierarchical relation between the current executing instance (super-instance) and an instance of another module (sub-instance). The hierarchical invocation is accompanied by the execution of the Statecharts of the sub-instance. The established hierarchical relation is dynamic in the sense that it is destroyed when the invoking state is deactivated. During an invocation, the sub-instance has access to the attributes of the super-instance.

III. HIERARCHICAL CONTROL SYSTEM

This section presents the generic modules used for the specification of the sequential controller. The attributes of these modules describe training parameters as well as the parameters related to control laws and to patients. Each of the following sub-sections is dedicated to the specification of one of the three levels of the sequential controller.

A. Level 1: Training session

The module (Fig. 3) corresponding to a training session is used to co-ordinate the training (or rehabilitation) modes and the consecutive phases that form the session. The attributes, whose values are to be instantiated for each specific training session, are: i) the number of phases of the session, *p*; ii) a table, *phase*, referencing the successive phases of the session; and iii) an index, *i*, to the current training phase.

The behaviour of this module is given by the states: *Preparation*, *Calibration*, *Phase*, *Abnormal_state*, and *Emergency_state*. A *start* order, given by the user, activates the *Preparation* state, which is used to initialise training parameters and to adapt the seat positioning to the morphology of the current user. The *Calibration* state can then be activated when the *calibrate* user order is given. This state invokes the *Movement_calibration* module, which is used to identify the gravity force of the limb and the moving part of the machine, for each of the positions of the whole range of movement. During the subsequent training phases, this data will be subtracted from the measured force to obtain a correct estimation of the muscular effort of the patient. The two modules referenced above (*Setup* and *Movement_calibration*) correspond to the execution of a number of successive operations, and their realisation depends on the structure of the target machine.

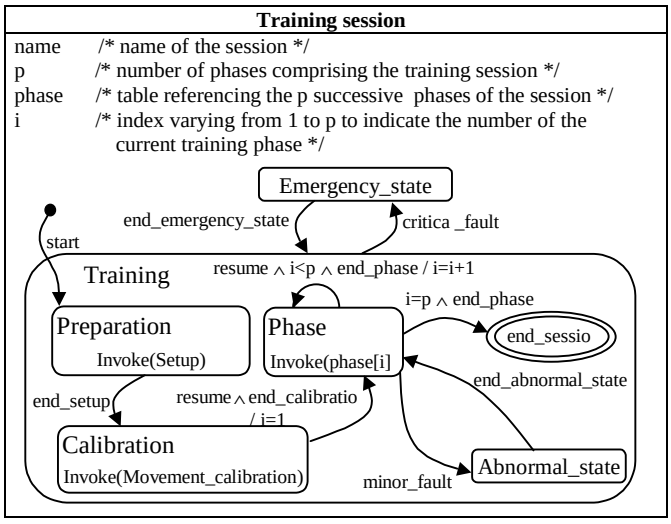


Fig. 3. Specification of the *Training-session* module

When the *Calibration* state ends, the *resume* user-order results in the initialization of the attribute *i* to 1, and the activation of the state *Phase*. This invokes the instance designated b *phase[i]* of the module *Phase* (corresponding to the second level of the hierarchy) to execute the first training phase. When a phase ends, a subsequent *resume* order triggers the self-looping transition, provided that the current phase has terminated (the corresponding instance has reached the state *end_phase*, see Fig. 4), and that it is not the final phase of the configuration (i.e., $i < p$). This transition increments the variable *i* so as to execute (invoke) the next phase of the training session. At the end of the final phase, $i=p \wedge$ *end_phase*, the state *end-session* is activated to signal the end of the training session.

The occurrence of serious fault, such as emergency_stop, end of course, or electronic errors, deactivates the *Training* super-state and activates the *Emergency_state*. These critical faults require an acknowledgement and a re-initialisation of the system. On the other hand, utilisation faults, such as the absence of the limb, generate an abnormal-fault event to stop the current movement and guarantee the user's safety. The machine should therefore be equipped by sensors to detect this type of fault.

B. Level 2: Phase

A phase is given by a succession of training series separated by a period of muscular relaxation. Each series comprises a number of repetitions of a particular upward movement pattern followed by a particular downward pattern, which are selected as a function of the muscular group, the training mode, and the training type required for the current phase. This information, to be given by a physiotherapist, is represented, respectively, by the attributes *muscle*, *mode* and *type* of the module *Phase* (Fig. 4). The other attributes of this module represent:

- the number of series, *nb_series*, and repetitions, *nb_rep*, of the phase
- pointers to the current series, *j*, and the current repetition, *k*;
- identifiers to the upward and downward patterns to be invoked in the current phase.

The Statechart of the module *Phase* has a super-state, *Movement*, during which the control laws are executed, a state *Relaxation* and a state *end_phase* indicating the end of the current phase. The invocation of an instance of this module (from the state *Phase* of module *Session*), triggers the initialization transition, which sets the attributes *j* and *k* to indicate the first repetition of the first series. This transition also results in executing the method *calculate_pattern*, which determines the modules of level three corresponding to the (upward and downward) movement patterns required for the current phase. The names of these modules are assigned to the attributes *upward_pattern* and *downward-pattern* as a function of the values of the attributes *mode*, *muscle* and *type* of the instantiated phase.

The state *Reach_reference_position* invokes the pattern named *Free_down* (not described in this paper) of level 3, which brings the leg to the low reference position of the patient. In this position, the first series of the current training phase starts executing the alternating states *Upward_movement* and *Downward_movement* that invoke the modules referenced, respectively, by the attributes *upward_pattern* and *downward_pattern*. The transitions between these two states are taken when the machine arrives to the upper or to the lower reference position. The transition from *Downward_movement* to *Upward_movement* also updates the repetition counter by incrementing the attribute *j*. When all the repetitions of the current series are executed (i.e., *j=nb_rep*) the *Relaxation* state is activated. This state, which can also be entered if the user order *relax* is given during the *Movement* super-state, has two output transitions:

- the first transition is taken when the end of the phase is reached (*k=nb_series*) to activate the state *end_phase*;
- the second transition is activated by the *repeat* user-order to start the next series of the current phase by incrementing *k*, setting *j* to 1 and activating the *Downward_movement* state.

The characteristic parameters (force versus position) of the movement under execution should be visualised throughout the states *Upward_movement* and *Downward_movement*, so as to provide information related to the performance of the patient.

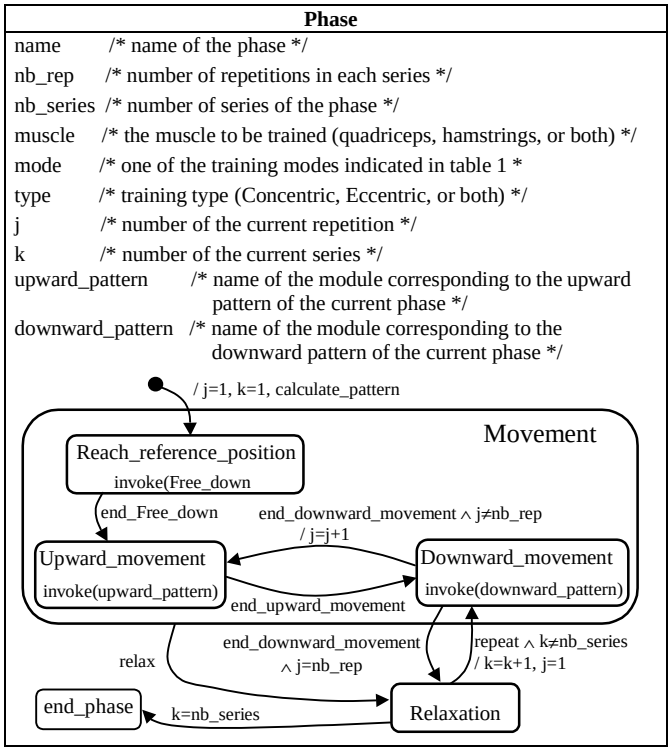
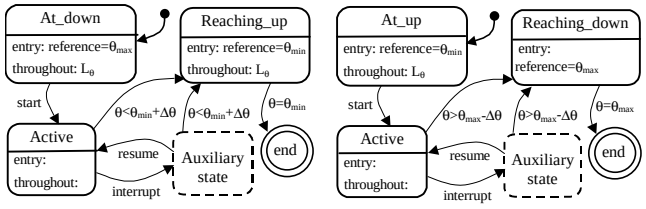


Fig. 4. Specification of the *Phase* module

C. Level 3: Upward and downward patterns

The modules of level 3 represent the states and the switching control sequences required to perform a give movement pattern. Thirty-six modules, corresponding to all possible movement patterns, were developed to drive the lower limb from one extreme angular position to the other. A generic structure of the dynamic behaviour of these modules was also identified. Figure 5a (respectively, 5b) depict this generic structure for the upward movements (respectively, downward movements). The upward movement pattern is discussed below (the behaviour of the downward pattern is similar). This pattern (Fig. 5a) comprises 5 basic states:

- an initial state, *At_down*, to keep the foot immobile at the lower extreme angular position, by executing a position control law with a reference θ_{max}, corresponding to the lower extreme angular position of the patient's limb.
- a progressive deceleration state, *Reaching_up*, to bring the machine to the upper extreme angular position, θ_{max}. This state is activated when the angular deceleration angle is reached, i.e., $\theta < \theta_{min}+\Delta\theta$.
- a terminal state, *end*, to stop the movement when the final position is attained.
- a main state, *Active*, that perform the main control law of the corresponding movement pattern (Table 1), subject to the required reference value. For the case of force and velocity control laws (i.e., isokinetic and isotonic modes), this state starts if the force applied by the patient exceeds the minimum working force. For the other modes, related to controlling the position of the angular articulation, this state starts if a predetermined position is attained.
- an auxiliary super-state, comprising one or many sub-state to execute the auxiliary operations required, together with the main control law, in view of guaranteeing some qualit and performance objectives related to the current training phase. On the one hand, this state is activated to stop the machine if a required medical constraint is not satisfied during the active state. In this case, the *interrupt* event (associated with the input transition) symbolises an insufficient muscular force, and the state, *Active*, can be resumed when the patient applies the required force value. On the other hand, the *Auxiliary* state may be used to drive the machine from one working position to another. In this case the event, *interrupt*, represents a temporal condition to limit the execution time of the *Activ* state.



a- upward movement b- downward movement

Fig. 5. Generic dynamic behaviour for the modules of level-3

For example, the upward pattern given by the *Isokinetic_up* module is depicted in Fig. 6. The specification of the other modules is given in [2]. The values of the attributes f_m, ω_m, θ_{min} and θ_{max} are to be specified by a physiotherapist. The Statecharts of the *Isokinetic_up* pattern give the switching sequence of control laws required to carry out an Isokinetic movement in extension. In this case, the machine is required to train the member with a constant-velocity upward-movement, providing that the patient's force exceeds the minimum required force, f_m. During each state, a position control law, L_θ, or a velocity control law, L_ω, is executed and an appropriate (position or velocity) reference is initialized. When the force applied by the patient, f_{est}, exceeds the minimum working force f_m, the state *Active* is entered to execute the velocity control law representing the main objective of an Isokinetic training phase. If the patient reduces the applied force below f_m, the state *Stop* is activated to bring the machine gradually to a stop by executing a zero-reference velocity control law. The state *Active* is entered again when the patient applies the required force value. The activity *compensation* is executed through all the states of this module to estimate the applied muscular force, f_{est}, on the basis of the force measured by the force sensor, as well as the information related to the gravity force.

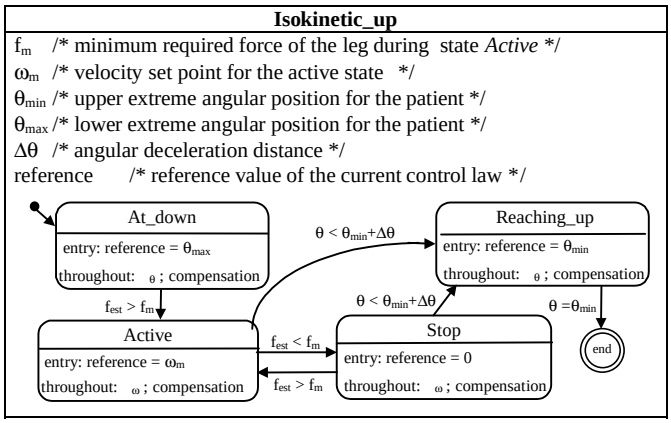


Fig. 6. Specification of the *Isokinetic_up* module

IV. Conclusion

This paper has proposed a generic formal model for sequential control of training machines for the lower limbs. This model can be instantiated to create a specific training configuration for the target machine. An object extension of Statecharts has been used to specify the generic model. This extension is based on a discrete model describing the co-ordinated tasks at the different hierarchical levels and allowing a coherent integration of continuous behaviour representing the switching control laws.

References

[1] M. Gross, "Intra-machine and inter-machine reliability of the Biodex and Cybex II for knee flexion and extension peak torque and angular work," *J. Orthopedic and Sports Physical Therapy*, vol. 13, pp. 329-330, 1991.

[2] S. Moughamir, *Conception et développement d'une machine d'entraînement et de rééducation des membres inférieurs.* PhD thesis, LAM-Univ. of Reims, 1999.

[3] J. Zaytoon, E. Richard, and L. Angelloz, "Spécification et conception d'un prototype asservi d'entraînement des membres inférieurs," *Innovation et Technologie en Biologie et Médecine*, vol. 17(1), pp. 57-70, 1996.

[4] J. Zaytoon, E. Richard, S. Moughamir, and L. Angelloz, " A formalism for complex control systems: Application to a machine for training and re-education of lower limbs," In *Proc. CESA'96 IMACS/IEEE Symposium on Discrete Events & Manufacturing Systems*. Lille-France, pp. 789-794.

[5] D. Harel, "Statecharts, a visual formalism for complex systems," *Journal of science of computer prog.*, vol. 8(1), pp. 231-274, 1987.