A Parametric Dataflow Model for the Speed and Distance Monitoring in Novel Train Control Systems

Benjamin Beichler, Thorsten Schulz, Christian Haubelt, Frank Golatowski

Institute of Applied Microelectronics and CE, University of Rostock,
{Benjamin.Beichler,Thorsten.Schulz,
Christian.Haubelt,Frank.Golatowski}@uni-rostock.de

Abstract. The Speed and Distance Monitoring (SaDM) in a train control system is a cyber physical system, which constantly has to process information about the train and its environment. The specification of such systems, however, is often done in an informal way, hindering formal analysis and optimization. In this paper, we propose to use Parametric Synchronous Dataflow Graphs (PSDF) to formally specify the SaDM. For this purpose, the information about the environment is modeled via piecewise constant functions, where each discontinuity corresponds to a physical location. As the number of relevant locations depends on the actual track side and, thus, is unknown a priori, we use parameters to construct consistent PSDF models. Based on our formal model, we have implemented the SaDM using *SCADE*.

Acknowledgement: This work was funded by the German Federal Ministry of Education and Research (Grant No. 01IS12021) in the context of the ITEA2 project openETCS.

1 Introduction

Model-based system engineering has proven to be a well-suited methodology to develop embedded systems and especially safety-critical cyber-physical systems. Model-based approaches are widely used in the automotive and avionics domain but are still uncommon in the railway sector. The increasing complexity of software in locomotive on-board units renders software development with traditional methods nearly impossible. We propose model-based engineering techniques as a means to ease this process. However, finding the right model for a model-based engineering approach is a challenging task.

The modeling formalism of *synchronous dataflow models* (SDF) and its extensions like *parametric synchronous dataflow* (PSDF) are well-suited for streaming applications e.g. from the domain of multimedia. The big advantages like well-developed formal methods for analysis and optimization could enhance the development of safety-relevant applications in other domains as e.g. the railway sector.

In this paper, we propose to use PSDF for modeling parts of the train control system, which constantly interacts with its physical environment. Train control systems (or respectively automatic train protection systems) have been developed since the very beginning of railway operation. Consequently, trains operated by different countries mostly use non-interoperable train control systems. Especially in the converging European Union this leads to a problem: all trains that need to cross borders also need to be equipped with several expensive train control systems.

The European Train Control System (ETCS), designed in the early 1990s, is the designated solution to overcome this problem within the European borders. ETCS includes a set of modern concepts for train control to achieve high speed and high utilization of the rail. Besides this, ETCS aims to be flexible to address all requirements of the national railway operators. The resulting ETCS standard became rather complex and difficult to implement, since the standard is currently only available as natural, non-formal language document. This leads to high development costs and incompatible implementations of different vendors caused by ambiguities of the specification.

In this environment, the openETCS project was created with the goal of an open source implementation of the on-board unit software. To achieve this, model-based systems engineering methods are employed. In this paper we present our efforts to model and implement the Speed and Distance Monitoring (SaDM) component using PSDF, which is part of the ETCS standard.

2 Related Work

Since the first release of the ETCS standard, several publications examined different aspects of the ETCS specification. Many of them deal with real-time properties and reliability of the communication link between train and track-side equipment. In [11,12,17,10] Petri net extensions are used to investigate the functional properties and stochastic guarantees of the communication. Modeling and calculation of SaDM of ETCS were covered in [16,8,14]. These focus on the functional properties of the computation and use of an application-specific modeling methodology. Other publications in the ETCS context focus on formalization and safety analysis. The authors in [7] show in three case studies how formal languages can ease the verification process of safety-critical systems. They show how the SPARK language and its toolset can be integrated into the existing development process to decrease the effort of system certification in the railway domain. However none of these publications deal with the modeling of the tight interaction with the physical environment.

In the last decades, the formalism of dataflow graph models as a refinement of process networks, have evolved to a valuable approach to develop streaming application like multimedia processing. The specialized type of *synchronous dataflow graphs* (SDF graphs) were presented in [13]. Due to their static nature, many analysis and optimization methods are available for SDF graphs. Since the expressiveness of SDF graphs is limited, many adoptions to increase their computational power have been proposed. Examples are *Boolean Dataflow Graphs* [6] and *cyclo-static dataflow graphs*[5]. *Parametric synchronous dataflow graphs* (PSDF graphs) extend the modeling features towards even more dynamic behavior, which allows reconfiguration of subgraphs based on a set of parameters. The computation of these parameters could be done by a configuration dataflow model or a parent model, where this subgraph is embedded. The applications of PSDF graphs described in research are mostly limited to applications in the area of de-/encoding data. An exception is [9], which discusses an approach to integrate the timing of cyber physical systems into process networks, but it lacks of other physical constrains as locations. In this paper, we propose to model the physical environment by a set of piecewise constant function, where each discontinuity corresponds to a physical location. As the number of discontinuities is not known a priori, we use the PSDF model ton construct a consistent model.

3 Parametric Synchronous Dataflow

The basic formalism for PSDF graphs are SDF graphs. A SDF graph $G = (V, E, cons, prod, D)$ consists of a set of Vertices V, a set of edges $E \subseteq V \to V$, token consumption rates $cons : E \to \mathbb{N}$, token production rates $prod : E \to \mathbb{N}$, and a delay function $D : E \to \mathbb{N}_0$. The vertices are actors communicating data tokens over unbounded channels represented by edges, so every channel is annotated with the number $d(e)$ of tokens on it. In SDF graphs the consumption and production rates need to be static. An actor $v \in V$ can be fired if $\forall e = (\widetilde{v}, v) \in E : d(e) \geq cons(e)$. If actor v fires, it consumes $cons(e)$ token from each incoming edge $e = (\widetilde{v}, v) \in E$ and produces $prod(e)$ token on each outgoing edge $e = (v, \widetilde{v}) \in E$ A SDF graph is called consistent, if a non-trivial repetition vector γ could be found, which describes the number of activations (firings) of every actor to get into the same state (count of tokens on the channels) as in the initial situation. In PSDF graphs this description is extended by configurable consumption and production rates are specified by parameters, which represent a runtime determined integer consumption or production rate.

4 ETCS - Speed and Distance Monitoring

To illustrate our proposed modeling approach, we use the speed and distance monitoring (SaDM) from the European Train Control System (ETCS). The SaDM is described next. One of the main tasks of ETCS is to supervise the speed and position of a train to ensure that the train stays in the permitted speed ranges. Because of the low friction between steel wheels and rail and the relatively high mass of the train, the braking distance is very large compared to, e.g., automobiles. As a consequence driving on sight is limited to relative low speeds and for higher speeds technical assistant is needed.

An established approach to ensure the safe track operation cascaded signals and mutual exclusive track usage is used. The size of the track segments significantly effects the utilization and possible throughput and therefore the profitability of a track. Since the signal equipment is fixed at the track side, a customization for different rolling stock is effectively impossible. This becomes a serious problem if trains with significantly different maximum speeds and braking abilities are used on a track.

To prevent a human failure of the perception of safety-critical information, all modern train control systems must have an automatic intervention possibility for dangerous situations. More sophisticated train control systems like ETCS make usage of customized signaling with displays within the train cab. This "cab signaling" helps to customize the speed and distance limits for every train. The challenge of a calculation on the on-board unit of the train control system is to ensure the safe operation of the train.

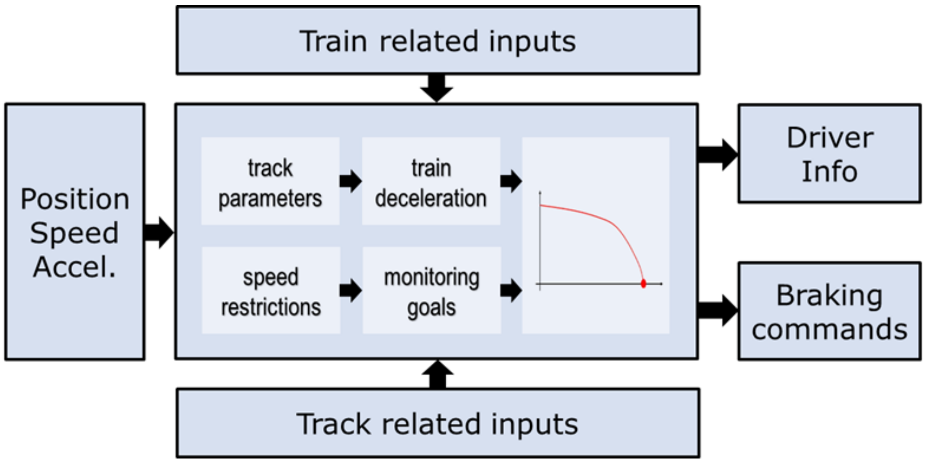


Fig. 1: Simplified Overview of the ETCS SaDM extracted from [15]

This includes the functional safety and the time-critical aspects of the calculation of speed and distance limits.

4.1 Overview

An overview SaDM is shown in Fig. 1. The tasks of SaDM are defined within the System Requirements Specification [15]. The main output of the SaDM comprises information for the driver, e.g., the currently permitted speed or monitoring targets. For critical situations, the SaDM issues automatic braking commands. In order to determine this information, SaDM needs several inputs such as dynamic values of the current position, speed and acceleration of the train. Moreover, a certain number of other train and track related inputs are needed which have lower dynamics as position or speed.

The most important train related inputs are the braking abilities of a train. Modern trains have multiple sets of brakes which have different operating principles and are used in several combinations according to various conditions. Thus, the applicable braking deceleration in a dangerous situation needs to be defined for all possible combinations. Other important characteristics such as curve tilt abilities, maximum train speed or the train length also need to be considered to calculate the train dependent impact on the speed and distance limits. All train related inputs are combined to a function called A_{safe}, that assigns a braking acceleration to the two independent parameters of speed and location on track. Hence, A_{safe} is a piecewise constant function or so-called step function of speed and position.

Beside the train characteristics, the track related information are important input data as well. A train equipped with ETCS receives information about the track properties while moving on it. This includes a profile of track slopes and a set of static speed restrictions which are caused by the shape of a track. Furthermore, dynamic speed restrictions (e.g., in areas which are under maintenance) are transmitted to the train. This collection of location-based speed restrictions is compressed to a single data structure called *Most Restrictive Speed Profile* (MRSP) which contains a single speed limit for every position on the track ahead. Again, the MRSP can be modeled by a piecewise constant function where every discontinuity corresponds to a location on the trackside.

From this profile the particular targets for the supervision are derived by getting all points with a decreasing allowed speed. An additional special target is derived from the limited permission of a train to move on the track. This *End of Authority* results from the *Movement Authority* which is transmitted by the chief of operation to the train. All of the described supervision targets are forwarded to the calculation of the target-specific braking curve. To predict the behavior of the train in an emergency case the *Emergency Brake Deceleration* (EBD) curve is one of the most important calculations. It is therefore in the focus of the following sections.

4.2 Emergency Brake Deceleration Curve Calculation

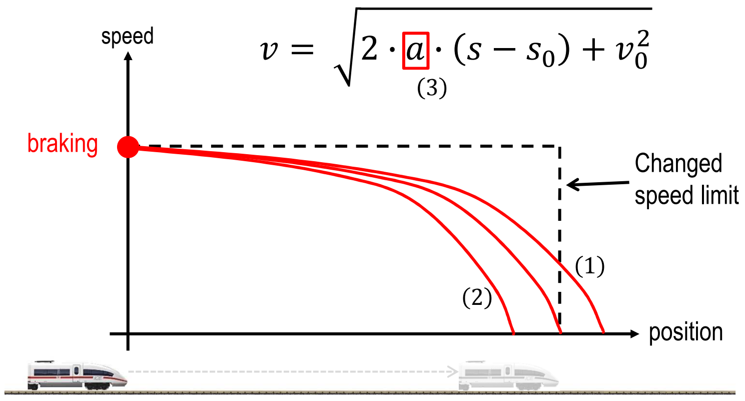


Fig. 2: Braking performance and its influence on the braking distance

The Emergency Brake Deceleration curve (EBD) represents the reliably expected braking behavior in case of emergency. The system has to use all available and allowed brakes to reach zero speed at a concrete location. In addition, there exist several constraints, e.g., there is a slippery track which leads to a reduced braking performance, or the system is unable to use all brakes but only a specific combination. The system has to calculate the position of brake initiation to stop before the target position under any circumstances. As shown in Figure 2 the braking performance influences the braking distance and as a consequence the maximum allowed speed for a constant deceleration value a at a given position s is described by the formula $v_{max} = \sqrt{2 \times a \times (s - s_0) + v_0^2}$, where s_0 and v_0 are a known point on the parabola. Since the deceleration value is only piecewise constant for a given speed and location range, several arcs of the form of the latter function are needed to describe the maximum allowed speed for a bigger part of the track. If the stop location and the braking performance on each section of the track are known, the latest point for brake initiation can be calculated to stop at the desired position. Hence, there is a need of a backward calculation algorithm which starts its calculation from the target location and calculates backwards to at least the current front end position of the train on the track as you could see in Fig. 3.

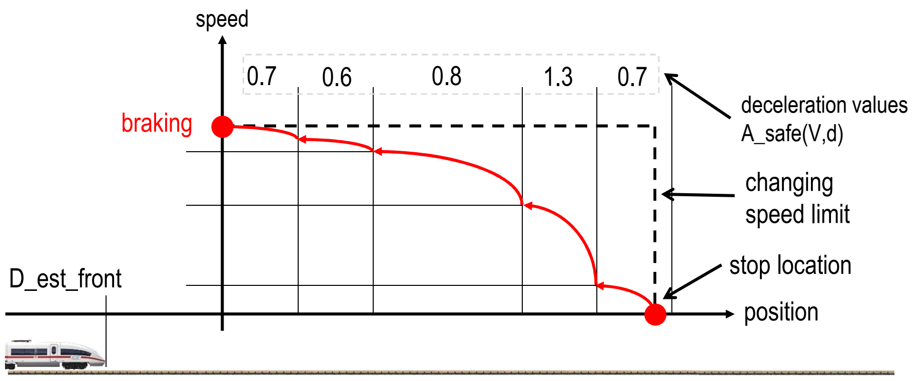


Fig. 3: Backward calculation of brake initiation depending on braking performance

The result of this algorithm is the maximum speed of the train on a specific position on the track. By exceeding this speed limit the train will fail to stop at the desired location. This information is known as EBD. After determining the maximum speed in comparison to the current speed, the ETCS on-board computer can intervene and brake automatically.

5 Parametric Dataflow Modeling of the EBD Calculation

For a formalized representation of the EBD calculation several analyses were done. The first step was the construction of the program flowchart in Fig. 4 to describe the algorithm. The calculation starts for every supervision target at the first known data point on the curve of allowed maximum speed which is the position of the supervised target itself and its associated speed limit. The initialization phase also includes a lookup into the two-dimensional array `A_safe(V,d)` containing information about deceleration values of the train depending on the track position and speed. These three values lead into a first arc of the EBD. Afterwards, the iteration checks whether the current iteration point is behind the current real estimated front end position of the train. This condition serves as a fast exit of the algorithm which is specified in [15] because the information behind the current real front end is mostly irrelevant. Following model shows that for

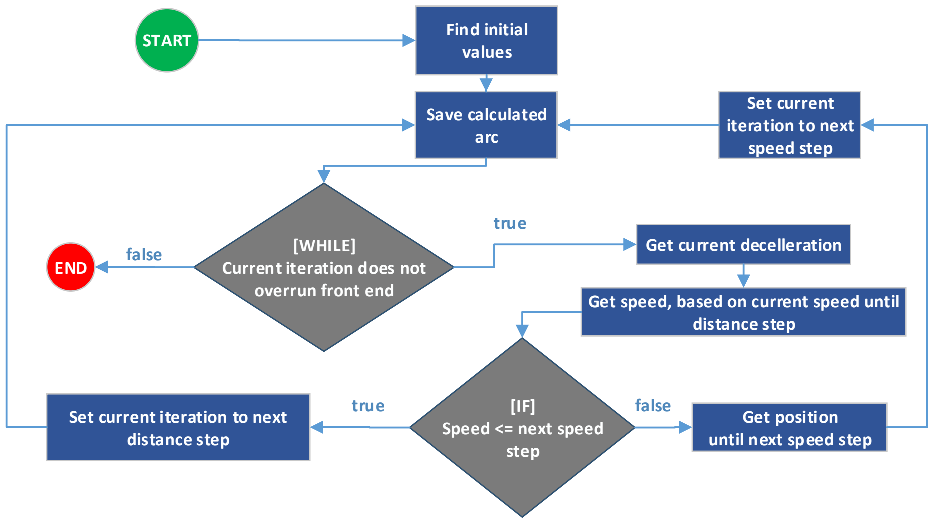

START
Find initial values
Save calculated arc
Set current iteration to next speed step
[WHILE] Current iteration does not overrun front end
false
END
true
Get current decelleration
Get speed, based on current speed until distance step
[IF] Speed <= next speed step
true
false
Set current iteration to next distance step
Get position until next speed step

Fig. 4: Algorithm of the EBD curve calculation

the worst case analysis and a static memory allocation this condition could be substituted by a static parameter-based condition.

The next step of the calculation determines the current acceleration of the `A_safe(V,d)` function and calculates the speed which the train would have at the next speed step if constantly accelerated with the current speed starting at the current position. This speed is compared to the speed of the next speed step of `A_safe(V,d)`. If it is lower, the arc is valid and could be saved and the current speed and position are updated to the end of the new arc. In the other case, the arc is only continuous until the speed of the next speed step of `A_safe(V,d)` so that the position where this speed step is passed needs to be determined. These three values make up the following arc and the current iteration values are updated.

For clarification it is noted that – seen from the absolute position – the calculated arc parameters are the end point of an arc and describe the parabolic before this position until the next point with other parameters is reached.

s[m] / v[m/s]	0	1000	3000	4400
0	1.0	1.2	1.3	0.7
40	0.8	0.6	0.8	0.9
80	0.7	0.7	0.6	0.8
120	0.6	0.8	0.6	0.7

Table 1: Iteration through `A_safe(V,d)`

Table 1 illustrates an example of an iteration through a simplified `A_safe(V,d)`. As already indicated by the parameter of the function, the table consists of a location and a velocity dimension. While iterating over this function, a diagonal path through this table is taken. The only possible directions are the cell at the left or the bottom cell. Special conditions could also allow a diagonal jump, but this could be safely approximated by two arcs, where the first one would start and end at the same point.

When we choose the parameter N to describe the number of position steps and the parameter M as the number of speed steps, the worst case of needed iteration steps is from the right uppermost corner to the left lowermost corner which will lead into a count of $N + M - 1$ iterations. Most safety-relevant software designs inhibit dynamical memory allocation. Thus, a result array with $N+M-1$ arcs entries needs to be reserved.

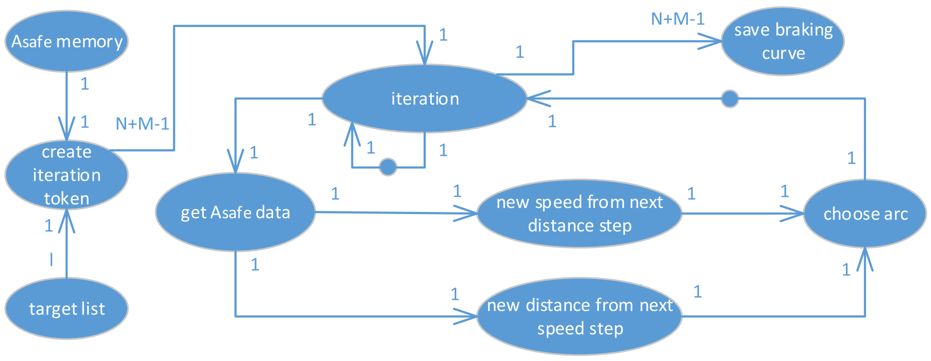


Fig. 5: Parametric Synchronous Dataflow of Braking Curve Calculation

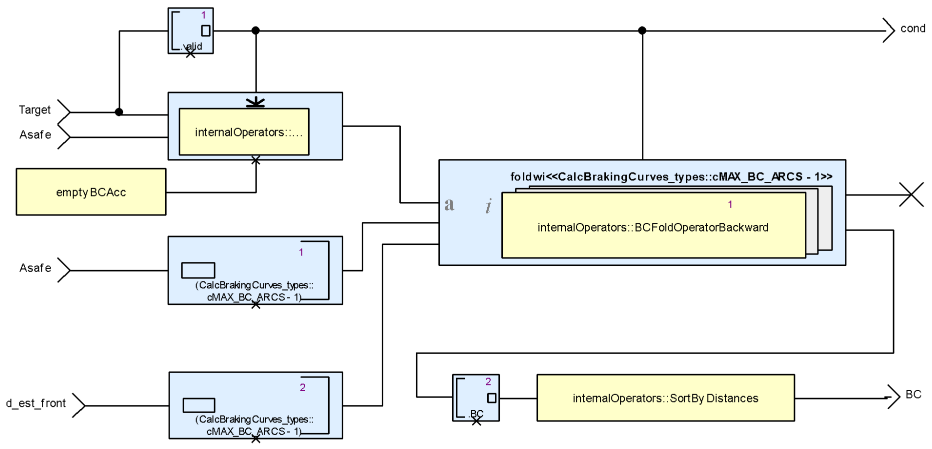
With this parameter model a parametric synchronous dataflow was implemented, which is shown in Fig. 5. The $N + M - 1$ factor is taken from the physical environment. Note that the exact value is depending on the track side and is unknown at compile time. Here a parametric model is needed. An additional aspect present in the parametric model, which was unmentioned in the previous model, is the effect of multiple targets. Since every supervision target needs its own adjusted emergency braking curve, the calculation has to be repeated for every target. Therefore, for every target $N + M - 1$ tokens are generated with the data of `A_safe(V,d)` and the specific target to match the number of iterations for every curve. Afterwards, the iteration node consumes one token from this edge and compares its value to the last iteration which is saved in the self-edge.

If they differ, a new braking curve is calculated and the first arc based on the target data is saved. In this case the token from `choose arc` is discarded, as it is a dummy trailing token of the last braking curve.

As in the program flow before (see Fig. 4), the cycle in the graph determines the end point of the next arc with its parameters. Because of the data dependency of each arc on the last iteration, low parallelism can be achieved, except the parallel calculation of the two possible cases. Until the $N+M-1$-th iteration, every result of the `choose arc` node is used as a new end point of an arc. But for the next braking curve a dummy token needs to be generated. In general, if the path through `A_safe` is shorter than $N+M-1$, e.g., if the target is not in the upper right corner, the last significant arc is in the left lower corner and all arcs to fill up the structure are copies of this arc. Therefore, the trailing token of the last iteration is such a copy as well.

Because of the sequential characteristics of this PSDF graph, the values of the repetition vector of the cyclic subgraph are $N+M-1$ firings for every target. If the count of targets is considered, the factor l needs to be multiplied. Since the spatial dimensions change slowly, for instance only if the train gets new information from the track side and every change results in a recalculation of all curves, the parameters are of the static type described in [4]. But the parameters N and M also met the requirements for the dynamic type of [4], because the subgraph fulfill the local synchrony condition defined in [4].

6 Implementation



1
.valid
cond
Target
Asafe
internalOperators::...
emptyBCAcc
foldwi<<CalcBrakingCurves_types::cMAX_BC_ARCS - 1>>
a
i
1
internalOperators::BCFoldOperatorBackward
Asafe
1
(CalcBrakingCurves_types::
cMAX_BC_ARCS - 1)
d_est_front
2
(CalcBrakingCurves_types::
cMAX_BC_ARCS - 1)
2
.BC
internalOperators::SortBy Distances
BC

Fig. 6: Top operator of calculation subgraph

The realization of the parametric SDF graph was done as part of the ecosystem of the openETCS project. Therefore, the integration into the existing modeling framework *SCADE-Suite* was used. *SCADE-Suite* is a widely used model development tool which is used to generate safety-critical software with the requirements of, e.g., *ISO 26262* or *EN 50128*. *SCADE-Suite* bases on the synchronous reactive language *scade*, a successor of the *esterel* language (first mentioned in [1], see [2,3] for further details). The translation of the given PSDF graph needs some adaptation, caused by the different models of computation. But the changes are minimal due to the nature of applying the PSDF graph. Since the cyclic subgraph of the computation only consists of edges which are obviously bounded to a capacity of 1, they could be easily translated into connections in *scade*. Moreover, synchronous reactive languages do forbid direct feedback loops. The loop of the calculation needs to be cut and feedbacked through a memory element or other elements. In the case of our graph, the feedback is solved by the iteration scheme `foldwi` operator which virtually performs a sequential instantiation of every iteration, where every iteration is connected through an accumulator connection. In Fig. 6 the top operator with the `foldwi` is shown. A special constraint of the virtual inflation of the `foldwi` operator is the fact that every input needs to have the dimensions of the iteration count, so they are inflated to `cMAX_BC_ARCS` which is equal to the former defined $N + M - 1$ parameter.

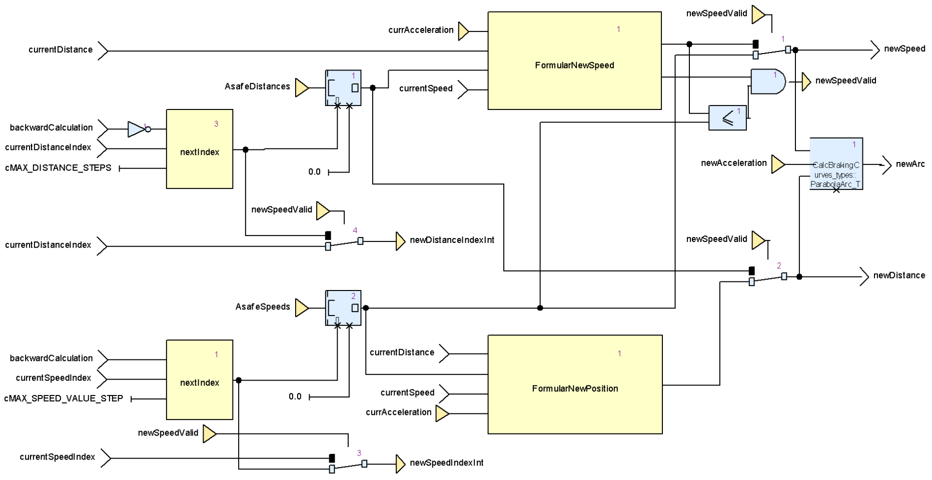


Fig. 7: Subgraph for inner operator calculation

Fig. 7 shows the inner part of the `foldwi` operator which defines the calculation cycle. The left side comprises the part of choosing the acceleration values according to the next distance step (upper part) and the next speed step (bottom part). Afterwards, the two cases are calculated in the boxes `FormularNewSpeed` and `FormularNewPosition`. The selection of the result of the two cases is broken up into several switches which are controlled by the Boolean expression in the upper right corner. The result is passed to the `newArc` output and saved in a higher stage.

What is unmentioned here is the operator to execute the braking curve calculation for every target of the target list. But again, this is only a `foldwi` operator with the target list as input.

7 Conclusion

This paper has presented a PSDF graph model of a real world safety-critical application of a cyber-physical system in the railway domain. The PSDF graph have proven the ability to reflect the spatial dimension parameters of the track side layout in the form of piecewise constant function, which were extensively used in ETCS. The created model of the calculation of the emergency brake deceleration curve have been implemented in the development environment *SCADE-Suite*, which encourage the goals of the openETCS project to get a formalized specification of the ETCS norm.

References

1. Berry, G., Moisan, S., Rigault, J.: Towards a synchronous and semantically sound high level language for real-time applications. In: IEEE Real Time Systems Symposium. pp. 30–40 (1983)
2. Berry, G.: The esterel v5 language primer version v5 91 (2000), ftp://ftp.inrialpes.fr/pub/synalp/reports/esterel-primer.pdf.gz
3. Berry, G., Gonthier, G.: The esterel synchronous programming language: Design, semantics, implementation. Science of computer programming 19(2), 87–152 (1992)
4. Bhattacharya, B., Bhattacharyya, S.: Parameterized dataflow modeling for DSP systems. IEEE Transactions on Signal Processing 49(10) (2001)
5. Bilsen, G., Engels, M., Lauwereins, R., Peperstraete, J.: Cycle-static dataflow. Signal Processing, IEEE Transactions on 44(2), 397–408 (Feb 1996)
6. Buck, J., Lee, E.: Scheduling dynamic dataflow graphs with bounded memory using the token flow model. In: Acoustics, Speech, and Signal Processing. ICASSP-93., 1993 IEEE International Conference on. vol. 1, pp. 429–432 (April 1993)
7. Dross, C., Efstathopoulos, P., Lesens, D., Mentré, D., Moy, Y.: Rail, space, security: Three case studies for SPARK 2014. Toulouse (Feb 2014)
8. Friman, B.: An algorithm for braking curve calculations in ertms train protection systems. Advanced Train Control Systems p. 65 (2010)
9. Grimm, C., Ou, J.: Unifying process networks for design of cyber physical systems. In: Electronic System Level Synthesis Conference (ESLsyn), 2011 (Jun 2011)
10. Hermanns, H., Jansen, D., Usenko, Y.: A comparative reliability analysis of etcs train radio communications (February 2005), aVACS Technical Report No. 2
11. zu Hörste, M., Hungar, H., Schnieder, E.: Modelling functionality of train control systems using petri nets. Towards a Formal Methods Body of Knowledge for Railway Control and Safety Systems p. 46 (2013)
12. Jansen, L., zu Hörste, M., Schnieder, E.: Technical issues in modelling the european train control system. Proceedings of the workshop on practical use of coloured Petri Nets and Design /CPN 1998 pp. 103–115 (1998)
13. Lee, E., Messerschmitt, D.: Synchronous data flow. Proceedings of the IEEE 75(9), 1235–1245 (Sept 1987)
14. Nitsch, A., Beichler, B., Golatowski, F., Haubelt, C.: Model-based systems engineering with matlab/simulink in the railway sector. Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen (MBMV), Chemnitz, Germany, In proceeding (2015)
15. UNISIG: SUBSET-026 – System Requirements Specification. SRS 3.3.0, ERA (2012)

16. Vincze, B., Tarnai, G.: Development and analysis of train brake curve calculation methods with complex simulation. Advances in Electrical and Electronic Engineering 5(1-2), 174–177 (2011)
17. Zimmermann, A., Hommel, G.: Towards modeling and evaluation of etcs real-time communication and operation. Journal of Systems and Software 77(1), 47–54 (2005)