

An Analysis of the Control Hierarchy Modelling of the **CMS Detector Control System**



Y.L.Hwong*, A.Racz, B.Beccati, C.Deldicque, C.Schwick, D.Gigi, E.Cano, E.Meschi, F.Glege, F.Meijers, H.Sakulin, J.A.Coarasa, J.F.Laurens, J.Gutleber, L.Orsini, M.Ciganek, M.Simon, M.Zanetti, R.Gomez-Reino, R.Moser, S.Cittolin (CERN) A.Meyer, D.Hatton, U.Behrens (DESY) D.Shpakov, H.Cheung, J.A.Lopez-Perez, K.Biery, R.K.Mommsen, V.O'Dell (Fermilab) A.S.Yoon, C.Loizides, C.Paus, F.Ma, G.Bauer, J.F.Serrano Margaleff, K.Sumorok (MIT) S.Erhan (UCLA) A.Petrucci, J.Branson, M.Pieri, M.Sani (UCSD) J.F.Groote, T.Willemse (TUE)

Introduction

The high level Detector Control System (DCS) of the CMS experiment is modelled using Finite State Machines (FSM), which cover the control application behaviours of all the sub-detectors and support services. DCS SUPERVISO 3D visualization SUB – DETECTORS CONTROLS GLOBAL SERVICES RACK COOLING CONTROL

Background In the CMS FSM model there is a state/command interface Each sub-detector group of the CMS experiment is responsible for between a parent and its children. Commands are passed from a the development of its own FSM tree control layer modelling their parent to its children and the states of the children are system. CMS will then integrate all these sub-trees into a single propagated to the parent. Two types of objects are defined in this FSM tree. hierarchy: The diversity in the DU CU DCS development philosophy of • Control Units (CUs): They different sub-detector groups are the logical units and the enormous amount of parameters to be monitored is • Device Units (DUs): They a fundamental aspect of a



The FSM tree of the whole CMS experiment consists of tens of thousands of nodes, which makes the implementation of a homogenous and consistent system a non-trivial matter.

The micro Common Representation Language 2 (mCRL2) analysis technique is being adopted to describe and analyze the CMS FSM system. Using its accompanying toolset, the FSM system of the RPC sub-detector has been analyzed and verified.



Architecture



The Joint COntrol Projects (JCOP) at CERN has chosen the SMI++ framework for the modelling of the FSM systems. Logically related objects are grouped into SMI++ domains; in which the objects are organized in a hierarchical structure and form a sub-system control.

The distribution and data exchange of this large system is handled by DIM (Distributed Information Management system), which is based on the client / server paradigm.



> default(id',s). del
) <>

Objective

To ensure a sound logic implementation throughout a system of such size, the investigation of some desired properties such as deadlock and endless-loop freedom requires an equally, if not more, complex mechanism.

The modelling and analysis of the CMS FSM system is a challenging task. The application of the mCRL2 toolset for this purpose is investigated.



Philosophy

mCRL2 is a specification language that can be used to specify and analyze the behaviour of large distributed systems and protocols. The language is supported by a toolset enabling simulation, visualisation, behavioural reduction and verification of software requirements.



Visualization

From the LPS, a Labelled Transition System (LTS) can be generated. A LTS is an explicit representation of the state space and can be visualized using interactive GUI tools. A sophisticated way of visualization is through 3D representation by employing a clustering technique to reduce the complexity of the image. The visualizations help in scrutinizing the model based on unexpected visual anomalies. It is possible to mark transitions and deadlocks for investigation.

,			RPC_chamberHVLVT_v2.bisim.lts - LTSView	4	Mark	×
ile <u>V</u> iev	<u>T</u> ools	<u>H</u> elp		🔿 No ma	arks	
				Mark d	deadlocks	
				O Mark s	states	
				O Mark t	transitions	
				Mark clu	uster if any state is ma	rked 🗅

Specification

A mCRL2 specification is a plain-text file containing a model in the mCRL2 language. It can be seen as a mathematical model of the system.

A.M.		今 ·江
	proc PARENT (id':ID, s:State, cl:ChildList, il:IDList)	1
	=	
	$(\#_{11} \geq 1) \rightarrow \text{sum s_c:State.}$	
	<pre>sum i:ID. (i in il) -> receive_state_P(i,s_c).PARENT(id',s,update_state(cl,i,s_c),remove(il,i))</pre>	
	+ (#i1 == 0)->	
A	(
11	(s == OFF) ->	
	(
	(any_in_state(cl,ERROR) any_in_state (cl, TRIPPED)) -> move_to_ERROR.send_state_TM(ERROR).PARENT	
	<pre><> ((any_in_state((filter_type(cl,HV)),RAMPING_UP)) (any_in_state((filter_type(cl,HV)),RAMPING_DOWN)</pre>	
	<pre> (all_in_state((filter_type(cl,LV)),ON) && all_in_state ((filter_type (cl, HV)), STANDBY)) -> move_</pre>	
	<> (all_in_state((filter_type(cl,HV)),ON) && any_in_state((filter_type(cl,LV)),ON)) -> move_to_ON.send_	
	<pre><> (all_in_state (cl,ON) && all_in_state ((filter_type(cl,T)),OK)) -> move_to_ON.send_state_TM(ON).PARE</pre>	
	<> default(id',s). delta	
) <>	
	(s == STANDBY) ->	
	(
	(any in state (cl,ERROR) any in state (cl,TRIPPED)) -> move to ERROR.send state TM(ERROR).PARENT	
	<pre></pre>	411
	<pre><> any in state((filter type(cl,HV)), RAMPING DOWN) -> move to RAMPING.send state TM(RAMPING).PARENT(id'</pre>	ATT
	<pre><> anv in state(cl,OFF) -> move to OFF.send state TM(OFF).PARENT(id',OFF,cl,il)</pre>	AXIV
	(2) (a) in state ((filter type (c) W)) (N) ff all in state ((filter type (c) W)) (N)) (-2) move to (N) set	(AANA)

mCRL2 specification of the RPC sub-detector.

mCRL2 micro Common Representation Language 2

-	se survey	the a prince and the second the second built fits similar	
🔳 XSim - R	PC_chamberHVLVT_v1.lp	ps 📃 🔍 🔍	
File Edit	Automation Options	Views Help	
Transitions		/s	
Action		State Change	
receive_c	ommand_(1, cOFF)	s3_PARENT := push(4, dc195, 1, OFF, [c(2, HV, OFF), c(3, HV, OFF), c(4, T, OK), c	
receive_c	ommand_(1, cON)	s3_PARENT := push(4, dc195, 1, OFF, [c(2, HV, OFF), c(3, HV, OFF), c(4, T, OK), c	
receive_c	ommand_(1, cSTANDBY	/) s3_PARENT := push(4, dc195, 1, OFF, [c(2, HV, OFF), c(3, HV, OFF), c(4, T, OK), c	
Paramet	er Value		
s3_PARE	VT push(1, do	c, 1, OFF, [c(2, HV, OFF), c(3, HV, OFF), c(4, T, OK), c(5, T, OK), c(6, LV, OFF), c(7,	234
24 700	MONITOR push1(1, 0), emptystack1)	
S31_10P			
s31_TOP s32_HV	push2(1, d	Jc253, [c(8, BM1, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2)	
s31_TOP s32_HV s33_HV	push2(1, d push2(1, d	dc253, [c(8, BM1, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2) jc644, [c(10, BM1, OFF), c(11, BM1, OFF)], [], dc645, 3, OFF, emptystack2)	
s31_TOP s32_HV s33_HV s34_T	push2(1, d push2(1, d push3(1, C	Jac253, [c(8, BM1, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2) Jac644, [c(10, BM1, OFF), c(11, BM1, OFF)], [], dc645, 3, OFF, emptystack2) DK, [c(16, BM1, OK), c(17, BM2, OK)], [], 4, dc1035, emptystack3)	Stand of
s31_TOP s32_HV s33_HV s34_T s35_T	push2(1, d push2(1, d push3(1, C push3(1, C	dc253, [c(8, BM1, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2) dc644, [c(10, BM1, OFF), c(11, BM1, OFF)], [], dc645, 3, OFF, emptystack2) OK, [c(16, BM1, OK), c(17, BM2, OK)], [], 4, dc1035, emptystack3) XK, [c(18, BM1, OK), c(19, BM2, OK)], [], 5, dc1076, emptystack3)	-
s31_TOP s32_HV s33_HV s34_T s35_T s36_LV	push2(1, d push2(1, d push3(1, C push3(1, C push3(1, C push2(1, d	dc253, [c(8, BM1, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2) dc644, [c(10, BM1, OFF), c(11, BM1, OFF)], [], dc645, 3, OFF, emptystack2) DK, [c(16, BM1, OK), c(17, BM2, OK)], [], 4, dc1035, emptystack3) DK, [c(18, BM1, OK), c(19, BM2, OK)], [], 5, dc1076, emptystack3) dc1117, [c(12, BM2, OFF), c(13, BM2, OFF)], [], dc1118, 6, OFF, emptystack2)	
s3_TOP s32_HV s33_HV s34_T s35_T s36_LV s37_LV	push2(1, d push2(1, d push3(1, C push3(1, C push3(1, C push2(1, d push2(1, d	dc253, [c(8, BM1, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2) dc644, [c(10, BM1, OFF), c(11, BM1, OFF)], [], dc645, 3, OFF, emptystack2) OK, [c(16, BM1, OK), c(17, BM2, OK)], [], 4, dc1035, emptystack3) OK, [c(18, BM1, OK), c(19, BM2, OK)], [], 5, dc1076, emptystack3) dc1117, [c(12, BM2, OFF), c(13, BM2, OFF)], [], dc1118, 6, OFF, emptystack2) ic1278, [c(14, BM2, OFF), c(15, BM2, OFF)], [], dc1279, 7, OFF, emptystack2)	
s31_TOP s32_HV s33_HV s34_T s35_T s36_LV s37_LV s38_BOT	push2(1, d push2(1, d push3(1, C push3(1, C push2(1, d push2(1, d push2(1, d fOM_MO push4(1, C	dc253, [c(8, BMI, OFF), c(9, BMI, OFF)], [], dc254, 2, OFF, emptystack2) dc644, [c(10, BMI, OFF), c(11, BMI, OFF)], [], dc645, 3, OFF, emptystack2) DK, [c(16, BMI, OK), c(17, BM2, OK)], [], 4, dc1035, emptystack3) DK, [c(18, BM1, OK), c(19, BM2, OK)], [], 5, dc1076, emptystack3) dc1117, [c(12, BM2, OFF), c(13, BM2, OFF)], [], dc1118, 6, OFF, emptystack2) dc1278, [c(14, BM2, OFF), c(15, BM2, OFF)], [], dc1279, 7, OFF, emptystack2) JFF, dc1439, 8, [], [], emptystack4)	
s31_IOP s32_HV s33_HV s34_T s35_T s36_LV s37_LV s38_BOT s39_BOT	push2(1, d push2(1, d push3(1, C push3(1, C push2(1, d push2(1, d push2(1, d fOM_MO push4(1, C fOM_MO push4(1, C	dc253, [c(8, BMI, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2) dc644, [c(10, BM1, OFF), c(11, BM1, OFF)], [], dc645, 3, OFF, emptystack2) OK, [c(16, BM1, OK), c(17, BM2, OK)], [], 4, dc1035, emptystack3) OK, [c(18, BM1, OK), c(19, BM2, OK)], [], 5, dc1076, emptystack3) dc1117, [c(12, BM2, OFF), c(13, BM2, OFF)], [], dc1118, 6, OFF, emptystack2) dc1278, [c(14, BM2, OFF), c(15, BM2, OFF)], [], dc1279, 7, OFF, emptystack2) DFF, dc1398, 8], [], emptystack4) DFF, dc1439, 8], [], emptystack4)	
s31_IOP s32_HV s33_HV s34_T s35_T s36_LV s37_LV s38_BOT s39_BOT s310_BOT	push2(1, d push2(1, d push3(1, C push3(1, C push2(1, d push2(1, d push2(1, d push4(1, C TOM_MO push4(1, C TOM_MO push4(1, C	dc253, [c(8, BM1, OFF), c(9, BM1, OFF)], [], dc254, 2, OFF, emptystack2) dc644, [c(10, BM1, OFF), c(11, BM1, OFF)], [], dc645, 3, OFF, emptystack2) OK, [c(16, BM1, OK), c(17, BM2, OK)], [], 4, dc1035, emptystack3) OK, [c(18, BM1, OK), c(19, BM2, OK)], [], 5, dc1076, emptystack3) dc1117, [c(12, BM2, OFF), c(13, BM2, OFF)], [], dc1118, 6, OFF, emptystack2) dc1278, [c(14, BM2, OFF), c(13, BM2, OFF)], [], dc1279, 7, OFF, emptystack2) DFF, dc1439, 8, [], [], emptystack4) DFF, dc1449, 10, [], [], emptystack4)	

Linearization

The first step in the mCRL2 analysis process is to linearize the specification to obtain a Linear Process Specification (LPS). All that remains in a LPS is a series of condition – action – effect rules that specify how the system as a whole reacts to a certain stimuli.



LPSs are a compact symbolic representation of the state space of the specification. Most mCRL2 toolset operate on LPS rather than on state spaces.



 $P //Q \rightarrow a. (P //Q) + i. (P //Q_a) + i. (P //Q_b)$

 $+ a|i. (P // Q_a) + a|i. (P // Q_b)$

Model Checking

Model checking is a verification method to show that the system exhibits certain desired properties. Given a LPS and a formula (expressed in the *regular* modal µ-calculus), a Parameterized Boolean Equation System (PBES) is generated.

The model checking question of "does the formula hold for this LPS" is encoded in the PBES. By solving this PBES, an answer (true/false) to this question can be found.







lps2pbes

Conclusions

The mCRL2 is a versatile and powerful toolset for the study and modelling of large distributed system, as is seen by its adoption for the analysis and optimization of the CMS RPC FSM system. Further application on other parts of the system is planned and is envisaged to enhance the performance and provide a better insight into the whole CMS FSM system. However, proper training is required for an accurate modelling of the system in such a way that its properties can be efficiently checked using the toolset. As the toolset harbours great potential, an automated tool for the translation from the FSM to the mCRL2 language for verification purposes is foreseen to be a highly rewarding project for the future.

Acknowledgements

I would like to express my gratitude to Jan Friso Groote and Tim Willemse from the Design and Analysis of System group at the University of Technology Eindhoven for their generous help and support in the course of this research project.



* Marie-Curie Early Stage Researcher: yi.ling.hwong@cern.ch

This research project has been supported by a Marie Curie Initial Training letwork Fellowship of the European Community's Seventh Framework Programme under contract number (PITN-GA-2008-211801-ACEOLE)



Technische Universiteit This research project is a collaboration with the department of Computer Science of the of Technology Eindhoven University of Technology.