Robotics Domain Task Force Final Agenda ver.1.0.3 robotics/2012-03-01

OMG Technical Meeting - **Reston, VA, USA** -- March 19-23, 2012

		TF/SIG		http://robotics.omg.org/		
		Host	Joint (Invited)	Agenda Item	Purpose	Room
Sunday: WG activites(pm)						
13:00	17:00			Robotics DDC4RTC submitters meeting	Arrangement	Lake Anne A, 2nd FL
Monday: WG activity						
9:00	12:00			**DDC4RTC** (Robotic Infrastructure) WG(5h) - Noriaki Ando (AIST) and Seung-Woog Jung (ETRI)	discussion	Town Center, 2nd FL
				RoIS (Robotic Functional Services) WG(5h): - Su-Young Chi (ETRI), Koji Kamei (JARA/ATR) and Toshio Hori (AIST)	discussion	South Lakes, 2nd FL
12:00	13:00	**LUNCH**				Grand Ballroom E-G, 2nd F
13:00	18:00			Architecture Board Plenary		Lake Audubon, 2nd FL
13:00	18:00			**DDC4RTC** (Robotic Infrastructure) WG(5h) - Noriaki Ando (AIST) and Seung-Woog Jung (ETRI)	discussion	Town Center, 2nd FL
				RoIS (Robotic Functional Services) WG(5h): - Su-Young Chi (ETRI), Koji Kamei (JARA/ATR) and Toshio Hori (AIST)	discussion	South Lakes, 2nd FL
Tuesday: WG activity(am) and Robotics-DTF Plenary(pm)						
9:00	12:00			**DDC4RTC** (Robotic Infrastructure) WG(3h) - Noriaki Ando (AIST) and Seung-Woog Jung (ETRI)	discussion	Suite 1246, 12th FL
				RoIS (Robotic Functional Services) WG(3h): - Su-Young Chi (ETRI), Koji Kamei (JARA/ATR) and Toshio Hori (AIST)	discussion	South Lakes, 2nd FL
12:00	13:00	**LUNCH**				Grand Ballroom D, 2nd FL
15:00	15:30	Robotics		**Talk: A Trial Approach for Automation in Open Cut Mine - Takashi Tsubouchi (Univ. of Tsukuba)**	presenta ion and discussion	Lake Firfax B, 2nd FL
15:30	16:10	Robotics		WG Reports and Discussion (Service WG, Infrastructure WG, Models in Robotics WG)	presenta ion and discussion	
16:10	16:30	Robotics		Contact Reports: - Makoto Mizukawa(Shibaura-IT), and Young-Jo Cho(ETRI)	Information Exchange	
16:30	17:00	Robotics		Robotics-DTF Plenary Wrap-up Session (**DTF Co-Chair Election**, Roadmap and Next mee ing Agenda)	Robo ics plenary closing	
17:00				Adjourn joint plenary meeting		
17:00	17:30			Robotics WG Co-chairs Planning Session (Preliminary Agenda for next TM, Draft report for Friday)	planning for next meeting	
Wednesday: WG activitiy						
9:00	12:00			**DDC4RTC** (Robotic Infrastructure) WG(3h) - Noriaki Ando and Seung-Woog Jung	discussion	Town Center, 2nd FL
12:00	14:00	**LUNCH and OMG Plenary**				Grand Ballroom D, 2nd FL
15:00	15:45	**MARS**	**Robotics**	**DDC4RTC Revu Revised Submission** (Joint Plenary with MARS) - Noriaki Ando	Joint with **MARS**	**Lake Auduon B, 2nd FL**
14:00	18:00			**DDC4RTC** (Robotic Infrastructure) WG(4h) - Noriaki Ando and Seung-Woog Jung	discussion	Town Center, 2nd FL
18:00	20:00	**OMG Reception**				Grand Ballroom D, 2nd FL
Thursday: WG activitiy						
10:10	10:30	MARS	**Robotics**	Joint Plenary with MARS (tentative) (reserved for DDC4RTC RFP Re-Review and Voting)	Joint with **MARS**	**Lake Anne B, 2nd FL**
9:00	12:00			Robotics WG activity follow-up	discussion	South Lakes, 2nd FL
12:00	13:00	**LUNCH**				Grand Ballroom E-G, 2nd F
13:00	18:00			Architecture Board Plenary		Lake Audubon, 2nd FL
13:00	18:00			Robotics WG activity follow-up	discussion	South Lakes, 2nd FL
Friday						
8:30	12:00			AB, DTC, PTC		Regency AB, 2nd FL
12:00	13:00	**LUNCH**				Lake Anne B, 2nd FL

Other Meetings of Interest

Monday						
8:00	8:45	OMG		New Attendee Orientation		Lake Fairfax B, 2nd FL
9:00	12:00	OMG		Introduction to OMG Specifications Tutorial		Lake Fairfax B, 2nd FL
7:00	20:00	OMG		Cloud Standards Customoer Coouncil Meeting		Regency A, 2nd FL
Tuesday						
7:30	9:00	OMG		Liaison ABSC		Lake Thoreau, 2nd FL
8:00	17:30	OMG		Cloud Standards Customoer Coouncil Meeting		RegencyA, 2nd FL
17:00	18:00	OMG		RTF-FTF Chair's Workshop		Hunters Woods, 2nd FL
Wednesday						
8:30	17:30	OMG		Business Architecture Informa ion Day		Regency A, 2nd FL
14:00	15:00	OMG		DDS Interoperability Live Demonstration		Lake Anne B, 2nd FL
9:00	17:00	SysA		System Assurande PTF		Lake Fairfax B, 2nd FL
9:00	12:00	DDS		DDS-PSIG		Lake Anne B, 2nd FL
Thursday						
13:00	17:17	OMG		UPDM Tutorial		Lake Fairfax A, 2nd FL
9:00	17:00	SysA		System Assurande PTF		Town Center, 2nd FL
9:00	12:00	ManTIS		ManTIS DTF		Tall Oakes, 2nd FL

Please get the up-to-date version from http://staff.aist.go.jp/t.kotoku/omg/RoboticsAgenda.pdf

Minutes of the Robotics DTF Meeting
December 12-16, 2011
Santa Clara, CA, USA
(robotics/2012-03-02)

Meeting Highlights

- The deadline of the DDC4RTC revised submission was extended to the upcoming Reston Meeting.
- The final report of Robotic Technology Component (RTC-1.1) was accepted to issue.
- Three presentations;
  - "The Legal Aspects of Autonomous cars", Bryant Walker Smith (Stanford Univ.)
  - "Proteus: An ontology for experimental validation of solutions to robotic problems" , Laurent Rioux (THARES) [robotics/2011-12-04]
  - "Domestic Standardization Activity for Standardizing Voice Interface for Service Robots in Japan", Yosuke Matsusaka(AIST) [robotics/2011-12-05]

List of Generated Documents

robotics/2011-12-01 Final Agenda (Tetsuo Kotoku)
robotics/2011-12-02 Salt Lake City Meeting Minutes [approved] (Seung-woog Jung and Miki Sato)
robotics/2011-12-03 Opening Presentation (Tetsuo Kotoku)
robotics/2011-12-04 Proteus: An ontology for experimental validation of solutions to robotic problems (Laurent Rioux)
robotics/2011-12-05 Domestic Standardization Activity for Standardizing Voice Interface for Service Robots in Japan (Yosuke Matsusaka)
robotics/2011-12-06 List of Issues for RoIS Framework [Mon. version] （Toshio Hori)
robotics/2011-12-07 Robotic Functional Services WG Report (Toshio Hori)
robotics/2011-12-08 List of Issues for RoIS Framework [Tue. version] (Toshio Hori)
robotics/2011-12-09 Infrastructure WG Progress Report (Noriaki Ando)
robotics/2011-12-10 RTC1.1-RTF Report (Geoffrey Biggs)
robotics/2011-12-11 ISO/TC184/SC2 Contact Report (Su-Young Chi)
robotics/2011-12-12 IEEE/RAS Standardisation (Geoffrey Biggs)
robotics/2011-12-13 Wrap-up Presentation (Tetsuo Kotoku)
robotics/2011-12-14 Roadmap for Robotics Activities (Tetsuo Kotoku)
robotics/2011-12-15 Next Meeting Preliminary Agenda - DRAFT (Tetsuo Kotoku)
robotics/2011-12-16 Event, Repository, Directory Manager for DDC4RTC (Seung-woog Jung)
robotics/2011-12-17 Component Management Model and Target Data Model (Seung-woog Jung)
robotics/2011-12-18 List of Issues for RoIS Framework [Wed. version] (Toshio Hori)
robotics/2011-12-19 DDC4RTC Progress Report [mars2011-12-08] (Tetsuo Kotoku)
robotics/2011-12-20 DTC Report Presentation (Tetsuo Kotoku)
robotics/2011-12-21 Santa Clara Meeting Minutes - DRAFT (Miki Sato and Seung-woog Jung)

Robotic Technology Component RTF report	dtc/11-11-02
Specification with change bars	dtc/11-12-03
Specification without change bars	dtc/11-12-02
C++ header file	dtc/11-11-05
IDL file	dtc/11-12-04
XMI file	ptc/11-12-03
EAP files	ptc/11-12-04
Inventory	ptc/11-12-02

Minutes

Tuesday, 13 Dec, 2011, Bayshore East, 2nd Fl.
Robotics DTF Plenary Meeting
AIST, ETRI, JARA, UEC, Univ. of Tokyo, Univ. of Tsukuba (Quorums: 4)
18 attendees

10:55 - 11:00 Robotics DTF Opening Session, Chair: Dr. Kotoku

- Minutes takers: Seung-Woog Jung (ETRI) and Koji KAMEI (ATR)
- Salt Lake City Meeting minutes review
  approved : Univ. of Tokyo (motion), ETRI (second), Univ. of Tsukuba (white ballot)

11:00 - 12:00 Special Talk: Bryant Walker Smith (Stanford Univ.)

The legal aspects of autonomous cars

13:00 - 13:45 Special Talk: Laurent Rioux (THARES)

An ontology for experimental validation of solutions to robotics problems

13:45 - 14:30 Special Talk: Yosuke Matsusaka (AIST)

Domestic Standardization Activity for Standardizing Voice Interface for Service Robots in Japan

15:00 - 15:40 WG Reports

- Functional Service WG, Toshio Hori (AIST)
  . Review of issues & resolutions
    * 22 issues arising (15 from Kissimmee, then discussed in Tokyo)
  . Future schedule
    * Comments Due : 20th Feb, 2012
    * Report Due : 21st May, 2012
    * Report Deadline : 29th June, 2012
- Infrastructure WG, Noriaki Ando (AIST)
  . Merged submission has been postponed : Next March (Washington) meeting
  . Reviewed implementation by Shibaura-IT
    * CAN-open based RTC D&C platform
  . Reviewed merged submission
    * Component Data Model
    * Some diagrams are created and updated from this discussion
    * Sequence diagram will be added for readers' convenience
  . Port in DEPL and RTC is different
    * DEPL component model is based on CCM
    * RTC component model is based on UML component model
    * add additional features to DEPL for DDC4RTC
  . Compatibility of Components
    * interface level
    * port level
    * component level
  . ISO 19143 for describing query and constraints
- RTC 1.1 RTF, Geoffrey Biggs (AIST)
  . Comments from 3 sources
  . 8 resolved issues, 9 deferred issues, 1 duplicate

. AB comment : XMI exportable
- Deferred because it is API-breaking.
* should be handled in the next major version.

15:40 - 16:00 Contact reports

- ISO TC184/SC2 (Robots and robotic devices), Su-Young Chi (ETRI)
  - . Berlin meeting (2011/09)
    - * ISO/TC184/SC2/WG7 : Personal care safety
    - * ISO/TC184/SC2/WG1 : Vocabulary and characteristics
    - * ISO/TC184/SC2/WG8 : Service Robots
  - . Dates of next meetings
    - * Mtg #17 : Feb. 10(Fri), Orlando, USA
    - * Mtg #18 : July 13(Fri), Milano, Italy
- IEEE/RAS Standardization, Geoffrey Biggs (AIST)
  - . two areas (led by Raj Madhaven)
    - * Robot map data representation (P1873)
    - * Ontology
  - . Working Group accepted by the IEEE Standardization Association
    - * October, 2011
    - * Map data co-chairs : Wonpil Yu (ETRI), Geoffrey Biggs (AIST)
    - * Ontology chair : Craig Schlenoff
  - - Study group meeting at IROS 2011

16:00 - 16:30 Robotics DTF Wrap-up Session, Chair: Dr. Kotoku

- Robotics-DTF Co-Chair : postpone voting one more meeting
- Robotic Services WG Co-Chair
  - . Miki Sato (JARA/ATR) -> Koji Kamei (JARA/ATR)
  - . JARA (motion), AIST (second), ETRI (white ballot)
- Changing RoIS FTF chair requires vote (will be on Friday)
- Schedule for next meeting
  - - DDC4RTC revised submission

ATTENDEE (18attendees):

- Noriaki Ando (AIST)
- Geoffrey Biggs (AIST)
- Su-Young Chi (ETRI)
- Young-Jo Cho (ETRI)
- Julien Deantoni (INRIA)
- Miwako Doi (Univ. of Tokyo / Toshiba)
- Toshio Hori (AIST)
- InCheol Jeong (ETRI)
- Seung-Woog Jung (ETRI)
- Koji Kamei (JARA/ATR)
- Tetsuo Kotoku (AIST)
- Laurent Rioux (THARES)
- Yosuke Matsusaka (AIST)
- Shuichi Nishio (JARA/ATR)
- Takashi Suehiro (UEC)
- Takashi Tsubouchi (Univ. of Tsukuba)
- Miki Sato (JARA/ATR)
- Bryant Walker Smith (Stanford Univ.)

Prepared and submitted by Seung-Woog Jung (ETRI) and Koji Kamei (JARA/ATR)

Robotics-DTF Plenary Meeting
Opening Session

March 20, 2012

Reston, VA, USA

Hyatt Regency Reston

Approval of Minutes

Meeting Quorum : 3
AIST, ETRI, JARA, Univ. of Tsukuba,

Minutes taker(s):

- **Geoffrey Biggs**
- **Seung-Woog Jung**

Minutes review

Santa Clara Meeting Summary

Robotics Plenary: (18 participants)

– **3 Talks**
- “The Legal Aspects of Autonomous cars”, Bryant Walker Smith (Stanford Univ.)
- “Proteus: An ontology for experimental validation of solutions to robotic problems”, Laurent Rioux (THARES) [robotics/2011-12-04]
- “Domestic Standardization Activity for Standardizing Voice Interface for Service Robots in Japan”, Yosuke Matsusaka(AIST) [robotics/2011-12-05]

– **2 WG Reports**
- Robotic Infrastructure WG **[robotics/2011-12-09]**
- Robotic Functional Services WG **[robotics/2011-12-07]**

– The deadline of the DDC4RTC revised submission was extended to the upcoming Reston

– The final report of Robotic Technology Component (RTC-1.1) was accepted to issue.

Roadmap for Robotics Activities

robotics/2012-03-04

Item	Status	Santa Clara CA Dec-2011	Reston VA Mar-2012	Cambridge MA Jun-2012	Jacksonville CANCEL Sep-2012	Burlingame CA Dec-2012	? Mar-2013	POC / Comment
Flyer of Robotics-DTF [Publicity Sub-Committee]	Suspended							
Robot Interaction Service (RoIS) Framework RFP [Robotic Functional Services WG]	Done Jun-2011							
Dynamic Deployment and Configuration for RTC (DDC4RTC) RFP [Robotic Infrastructure WG] in MARS	In Process		Revised Submittion & Voting					Sponsor: MARS
Robotic Map Services RFP [Robotic Functional Services WG]	Planned							IEEE R&A?
etc…	Future							
Robotics Exhibition	Planned					Exhibition		
Robotics Information Day [Technology Showcase]	Future							
RoIS Finalization Task Force	In Process			FTF Report				
DDC4RTC Finalization Task Force	Planned		Charter				FTF Report	
RTC 1.2 or 2.0 Revision Task Force	Planned							
RLS 1.2 or 2.0 Revision Task Force	Planned							will go to ISO/TC211
Related Events					Chu-suk (Special Holidays in Korea)			

A Trial Approach for Automation in Open Cut Mine

Takashi Tsubouchi

YAMAZUMI Project

Intelligent Robot Laboratory
Department of Intelligent Interaction Technologies,
Graduate School of System and Information Engineering,
University of Tsukuba

robotics/2012-03-05

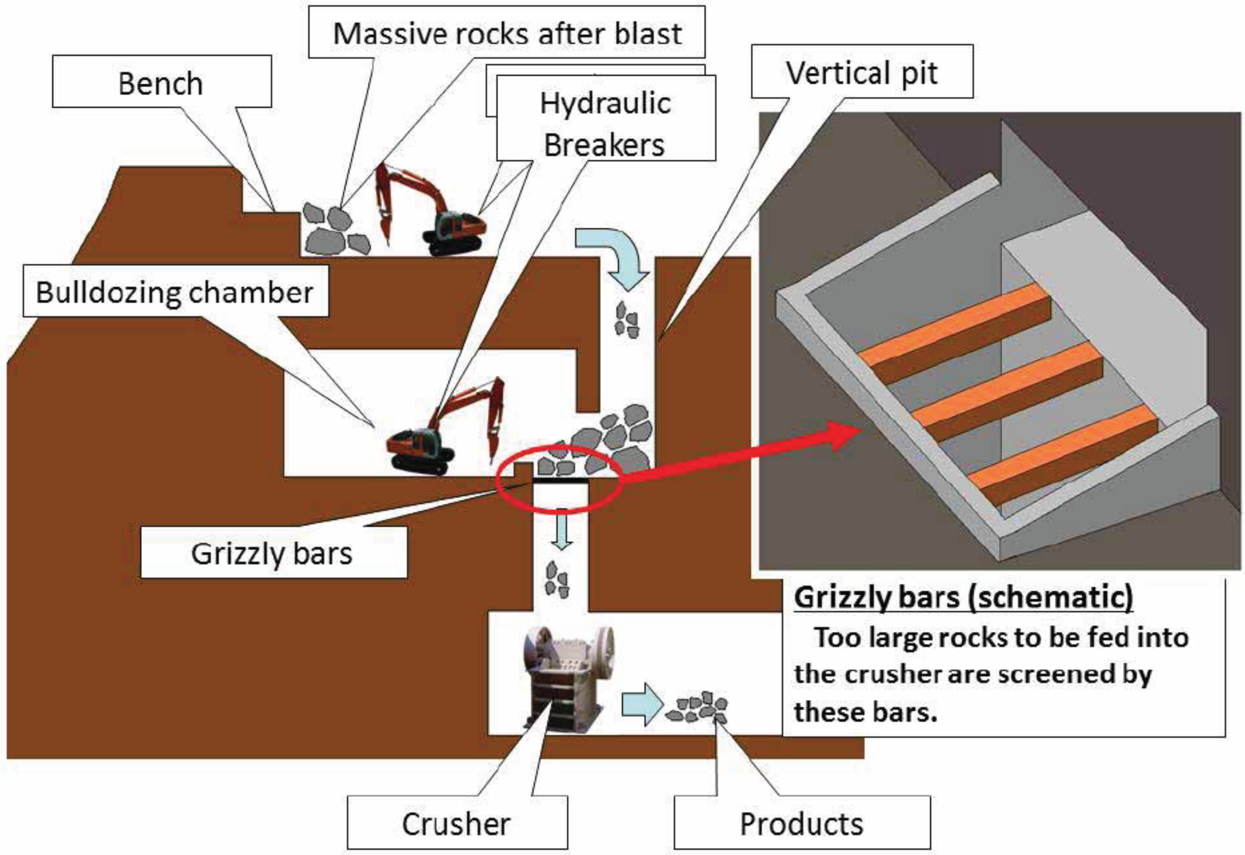
Backgrounds and Motivations

Open Cut Mine

- Limestone: Exportable resource from Japan
- Aggregate Production: Sedimentary rock or sandstone bedrock

An Open Cut Mine of Limestone

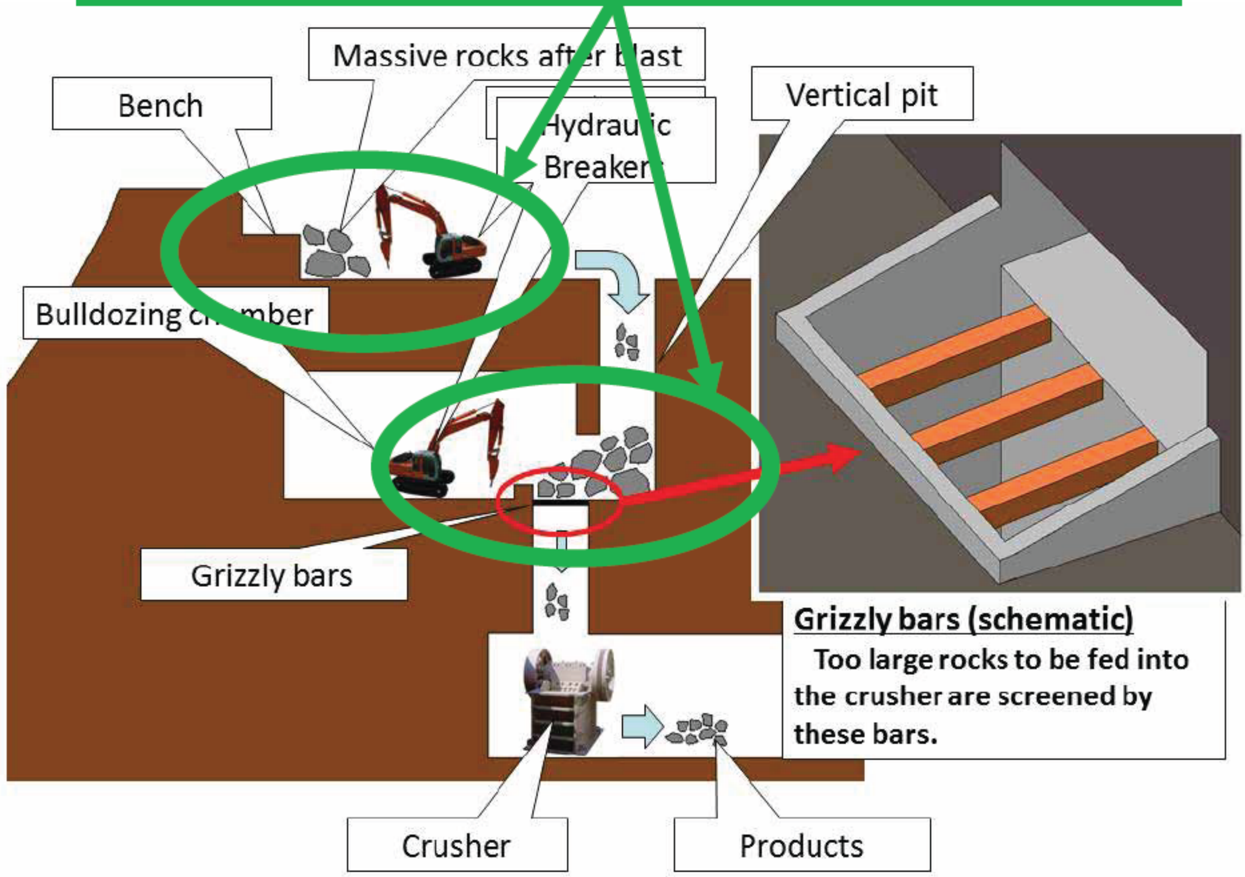
A Typical Schematic of Limestone Open Cut Mine



Grizzly bars (schematic)

Too large rocks to be fed into the crusher are screened by these bars.

Breaking Operation - Cobbing



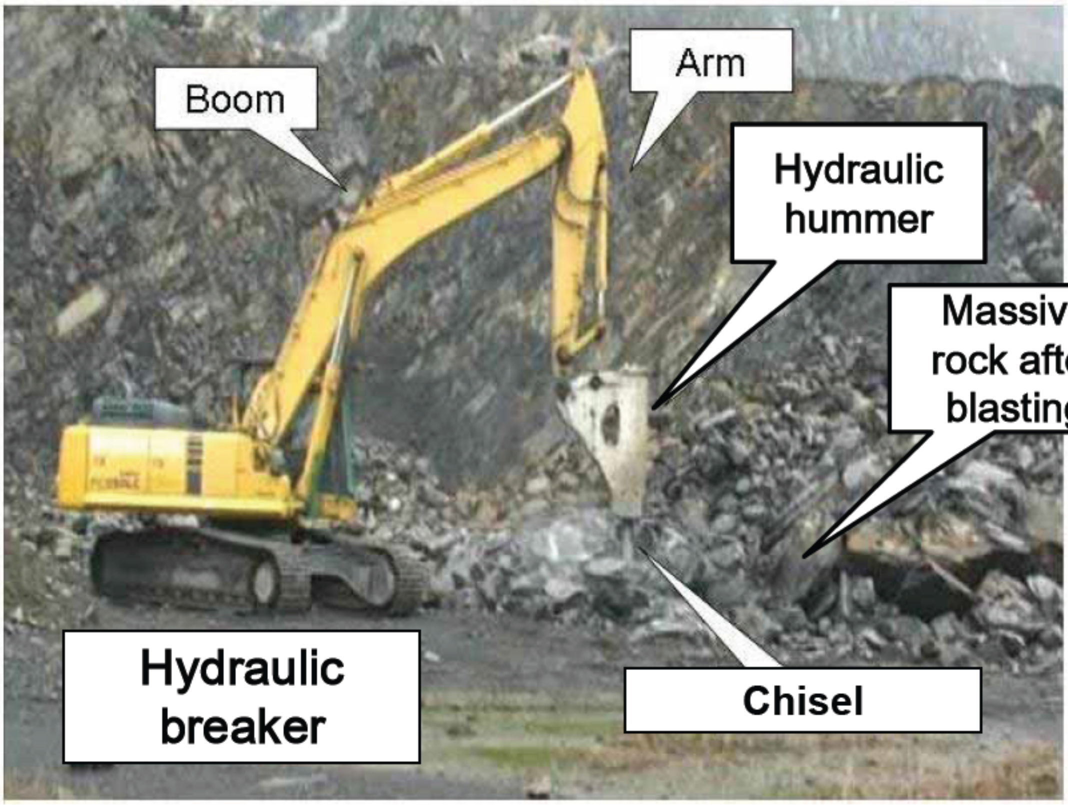
Grizzly bars (schematic)

Too large rocks to be fed into the crusher are screened by these bars.

Two Situations for the Breaking Operation

1. Breaking the massive rocks after blasting at the working front of the quarry.
   - ◆Onsite operation of the hydraulic breaker by a matured operator
2. Breaking the massive rock plugged at the grizzly bars in the bulldozing chamber
   - ◆Monitor the plugging at remote operation room
   - ◆Remote operation of the hydraulic breaker

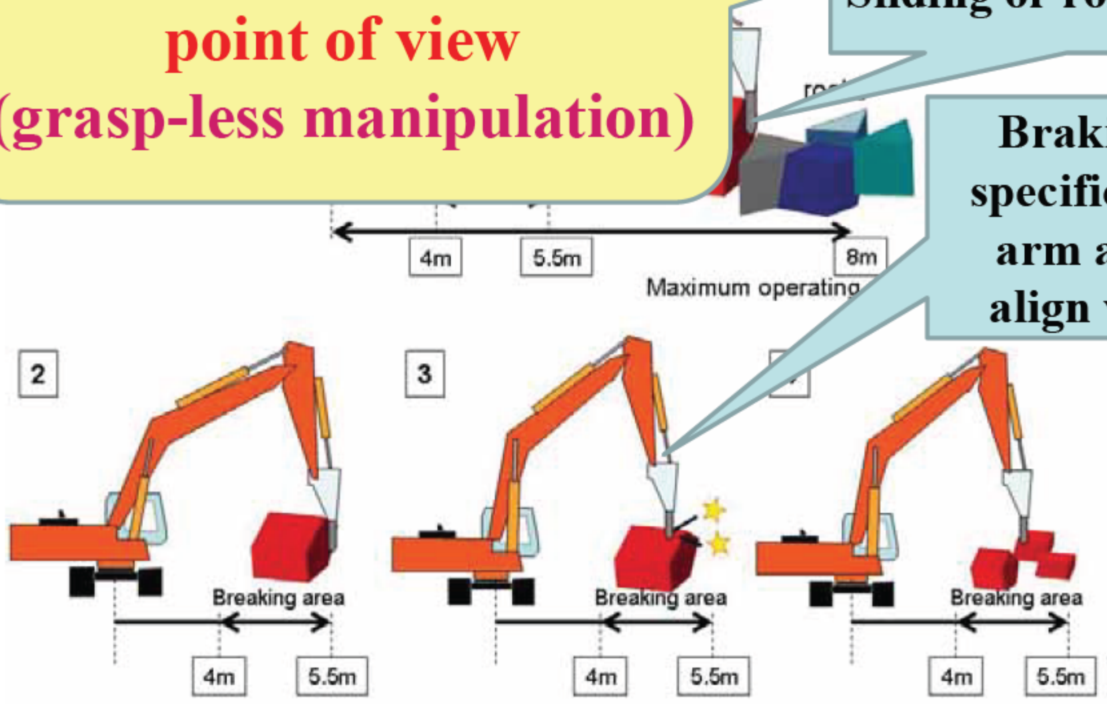
1. Breaking Operation at the Working Front



1. Breaking Operation at the Working Front

(movie)

1. Breaking Operation at the Working Front



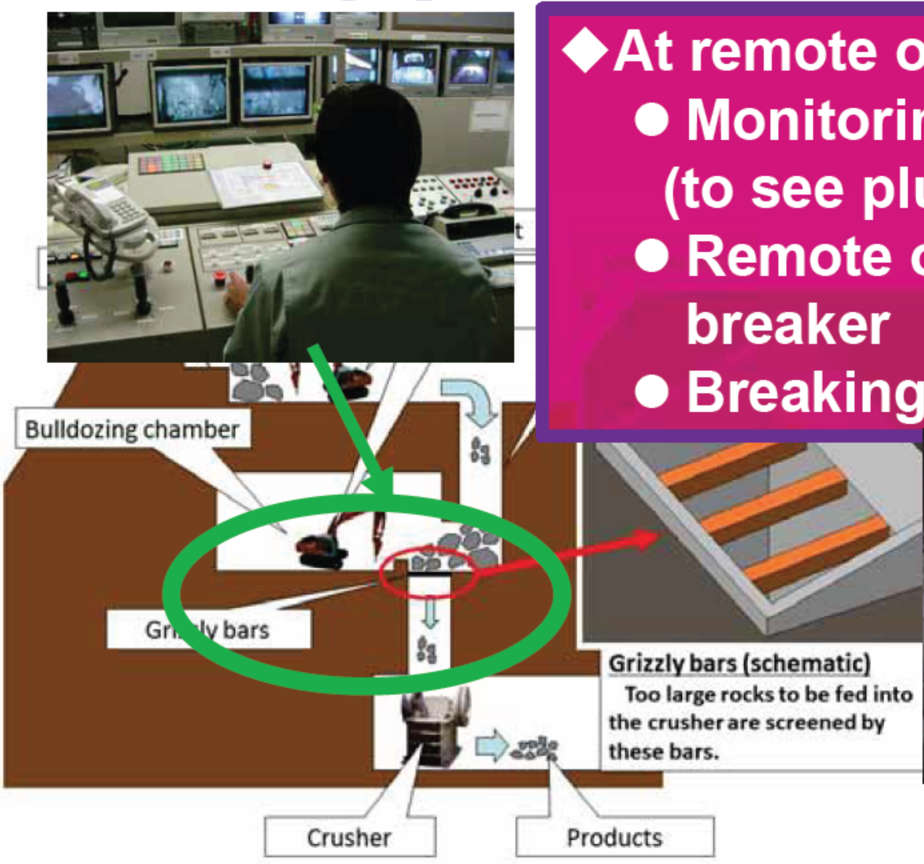
Interesting in the robotics point of view (grasp-less manipulation)

Operation of the chisel
Sliding or rolling the rock

Braking at the specific posture – arm and chisel align vertically.

2. Breaking operation in the bulldozing Chamber

◆At remote operation room
- Monitoring the chamber (to see plugging)
- Remote operation of the breaker
- Breaking the plugging rock



Grizzly bars (schematic)
Too large rocks to be fed into the crusher are screened by these bars.

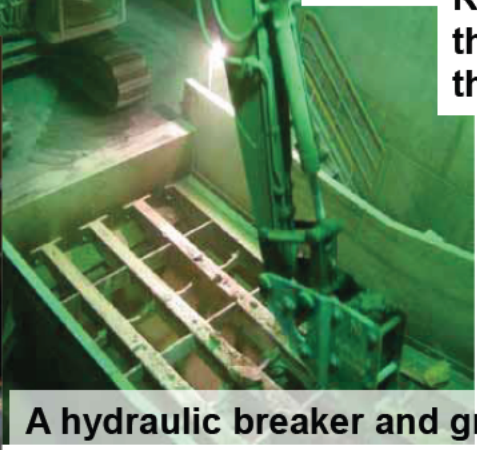
Grizzly bars and plugging

Inside of the Chamber

A Hydraulic breaker and a monitoring camera

Rocks moving through the grizzly bars from the vertical pit

A console in the remote operation room

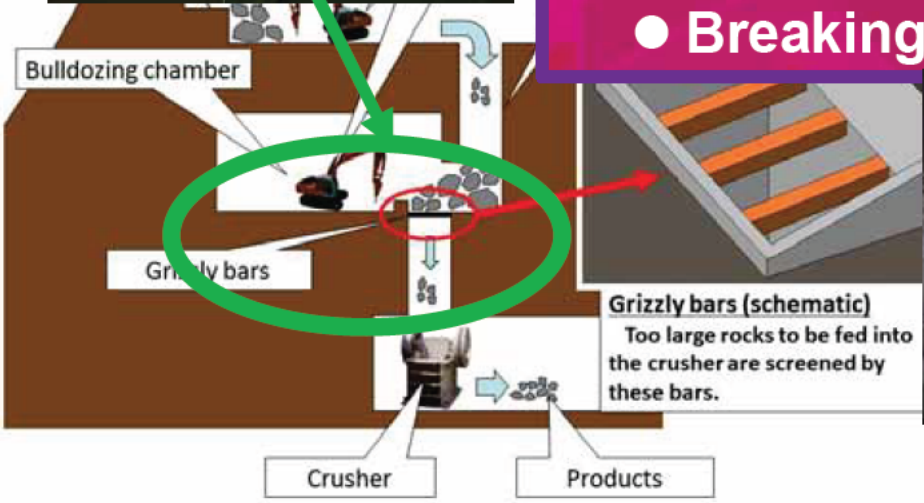


A hydraulic breaker and grizzly bars

2. Breaking operation in the bulldozing Chamber

◆At remote operation room

- ● Monitoring the chamber (to see plugging)
- ● Remote operation of the breaker
- ● Breaking the plugging rock



Too large rocks to be fed into the crusher are screened by these bars.

Grizzly bars and plugging

On Going Studies in the Lab

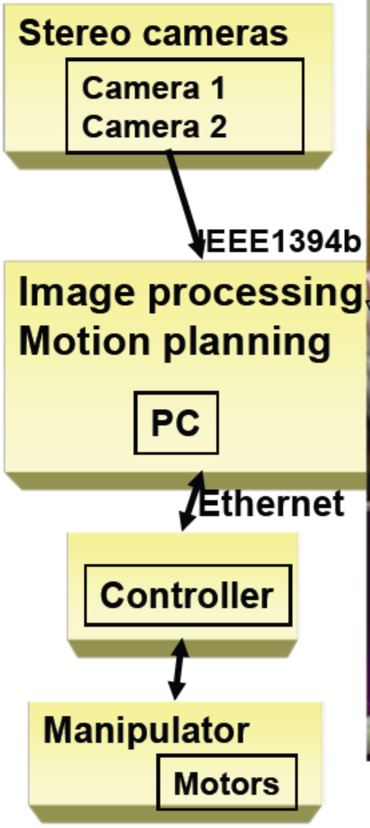
1. Breaking operation at the working front:
   - ◆1/12 scale electric powered manipulator + stereo vision
   - ◆Trial issue of automating rock sliding or rolling operation
2. Plugging monitoring at the chamber:
   - ◆Automatic detection of plugging by means of movie image processing (of recorded video images)

On Going Studies in the Lab

1. Breaking operation at the working front:
   - ◆1/12 scale electric powered manipulator + stereo vision
   - ◆Trial issue of automating rock sliding or rolling operation
2. Plugging monitoring at the chamber:
   - ◆Automatic detection of plugging by means of movie image processing (of recorded video images)

Trial Issue of Automating Rock Sliding or Rolling Operation

Lab Experiment Table



Stereo cameras

Camera 1
Camera 2

IEEE1394b

Image processing
Motion planning

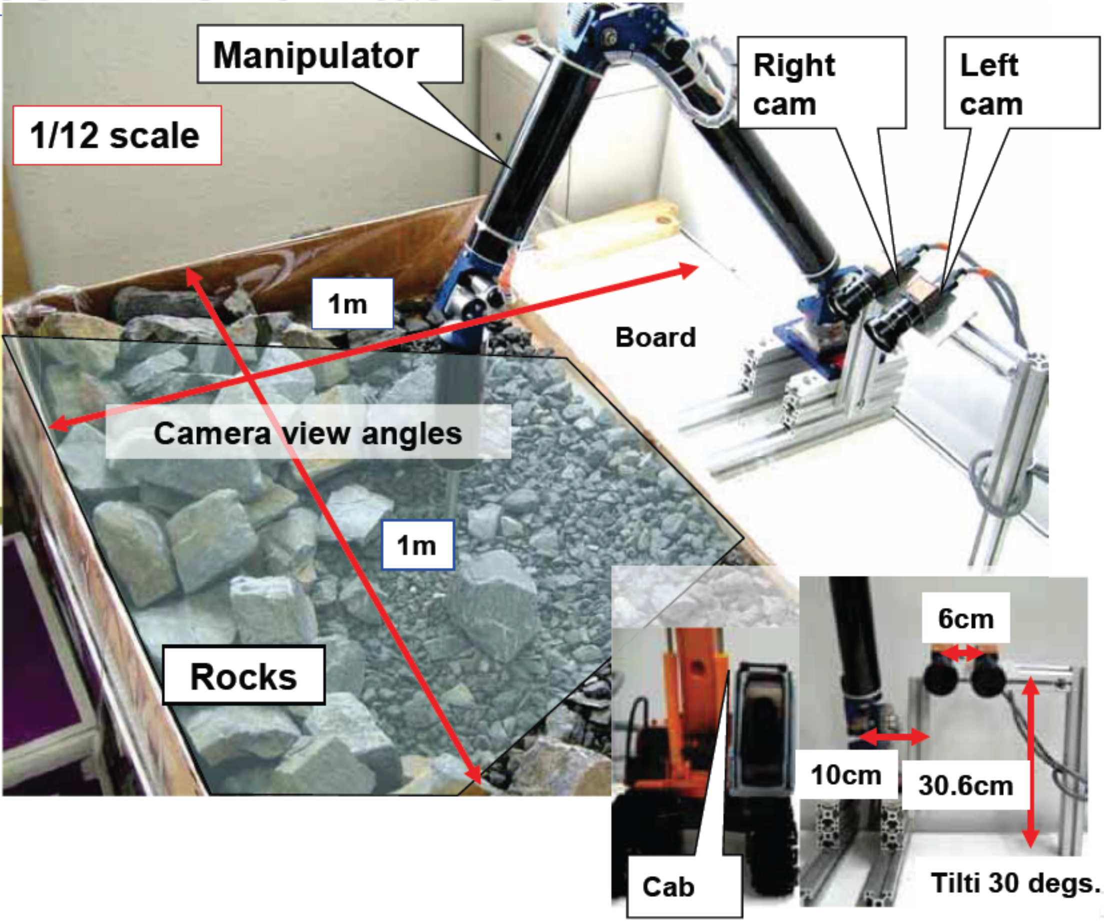
PC

Ethernet

Controller

Manipulator

Motors

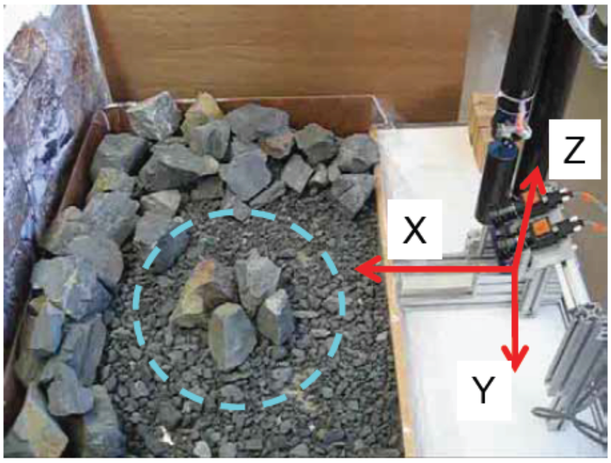


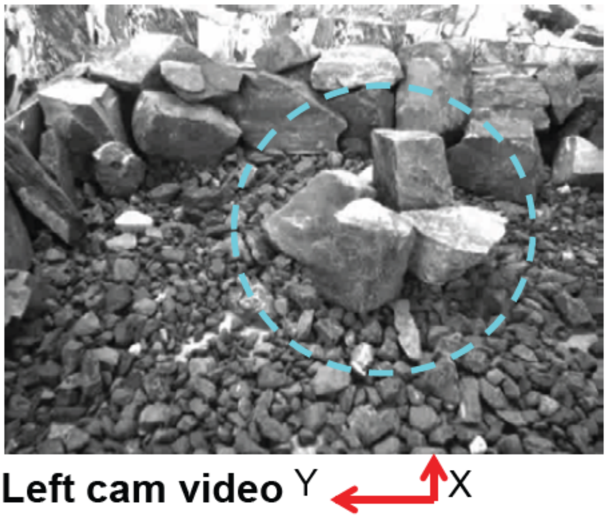
Rock Moving Experiment

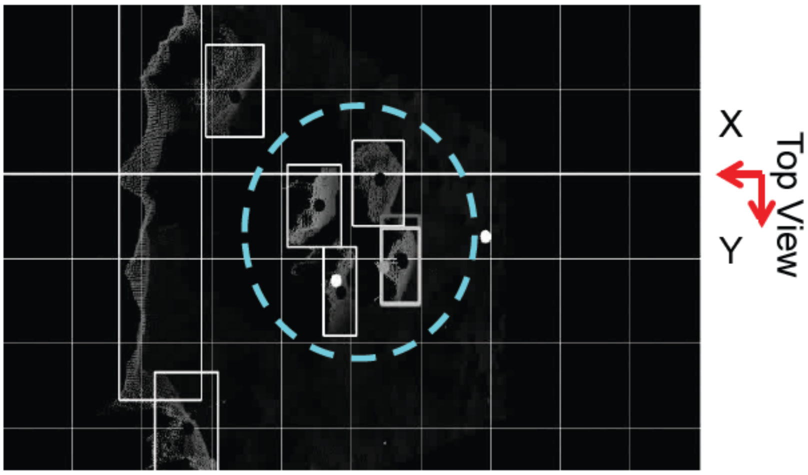
☐Moving 5 rocks

- 5 rocks are placed at around 50cm in X direction
- Move them within 37cm

Trial	30 times（29 success）
Ave. elapsed time	2m51s
Ave. trial times	20.66times
Ave. effector top speed	10cm/sec




Left cam video



Automated Rock Sliding or Rolling Operation - Now and Future -

- Now: Loop of “vision input and action” and the moving operation has been established and realized anyway.
- Future:
  - More precise recognition of the shape of the rocks
  - More matured motion planning according to the shapes and physical mechanics
  - Bridge to real machine experiment

On Going Studies in the Lab

1. **Breaking operation at the working front:**
   - ◆1/12 scale electric powered manipulator + stereo vision
   - ◆Trial issue of automating rock sliding or rolling operation
2. **Plugging monitoring at the chamber:**
   - ◆Automatic detection of plugging by means of movie image processing (of recorded video images)

Plugging Monitoring by Means of Video Image Processing

Automated Plugging Monitoring

1. Detection of rocks flow
   A) Optical flow
   B) Inter frame difference

2. Estimation of the plugging point

Detection of Rocks Flow

Optical flow

- Video image processing on PC is applied to recorded video images at the champber
- OpenCV library is utilized
- Red signal appears when the flow stops

Detection of Rocks Flow

Inter frame difference

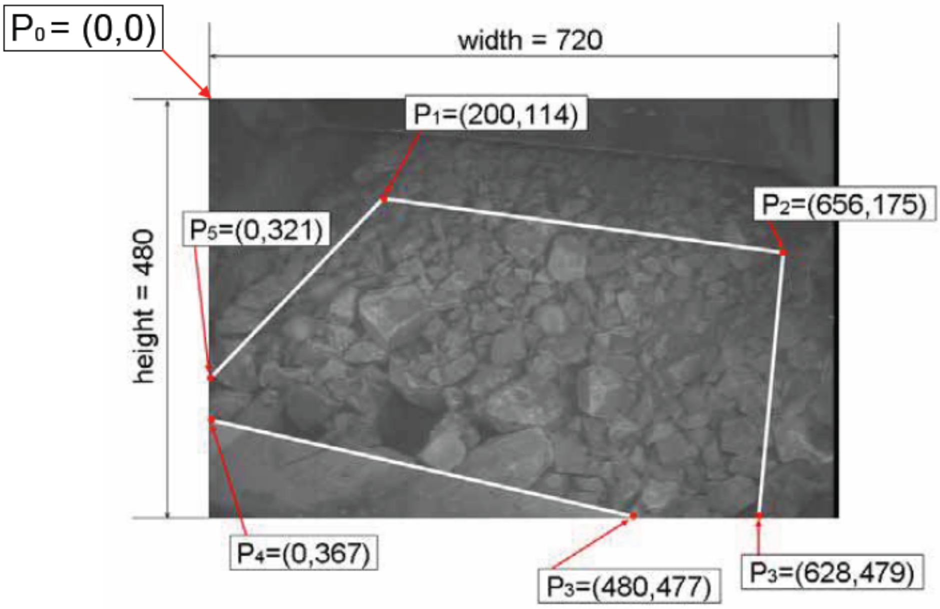
- Disadvantage of the optical flow
  - ➢ Flow stop judgment but slow flow

- Inter frame difference is also considered

The flow cannot be detected when the speed of the flow is low then significant optical flow will not appear.

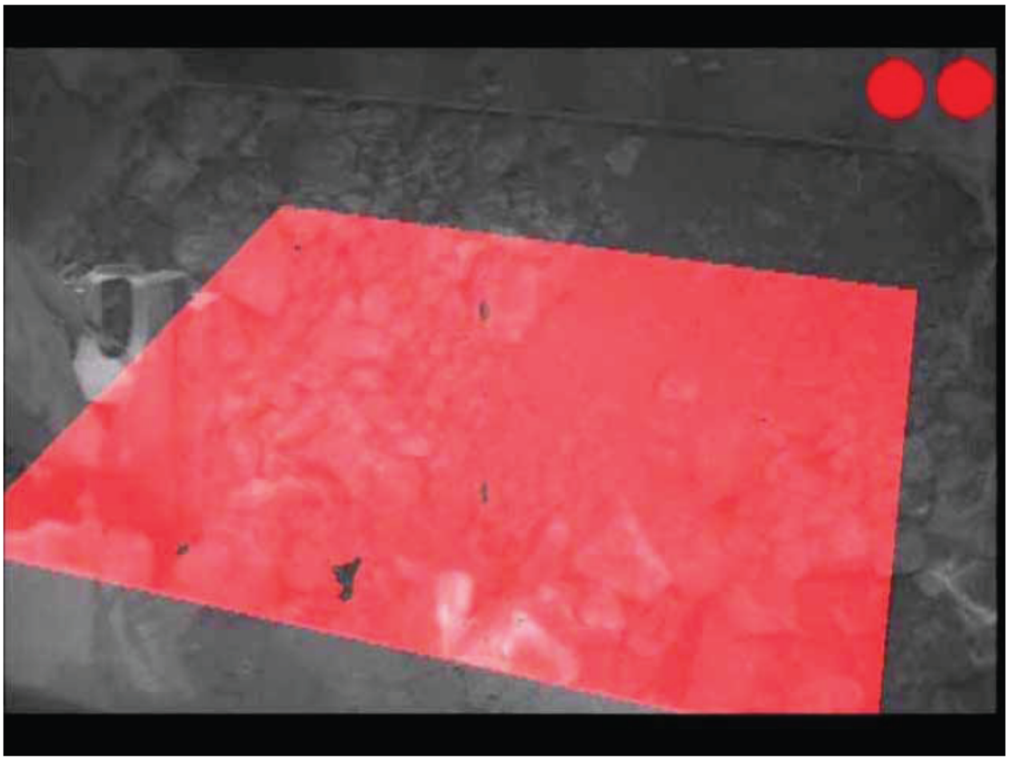
Rocks Flow Detection with Inter Frame Difference

- Detection Covered Area -



- Inter frame difference is calculated inside of the white rectangle.
- The white box is in the range of grizzly bars.

Rocks Flow Detection with Inter Frame Difference

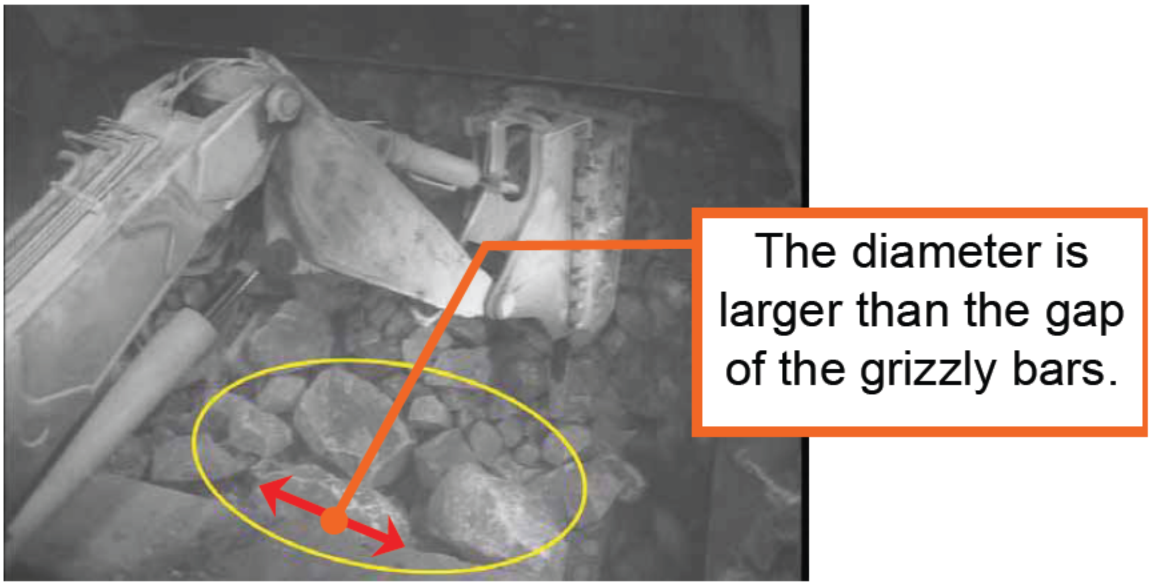


1. Get inter frame difference at every 5 frames
2. If no difference at a pixel, put red color on the pixel.
3. If red area becomes large (= the number of red pixels exceeds the threshold), the flow stops.

Locating the Plugging Rock

Locating the Plugged Rock
-Categorization-

The situation can be categorized into 3 categories.

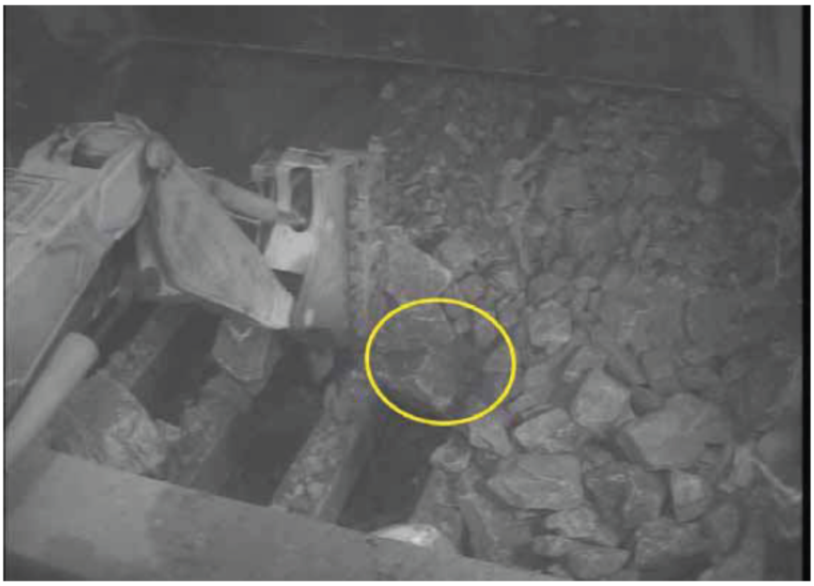


The diameter is larger than the gap of the grizzly bars.

Case1. Plugging by large rock

Locating the Plugged Rock
-Categorization-

The situation can be categorized into 3 categories.

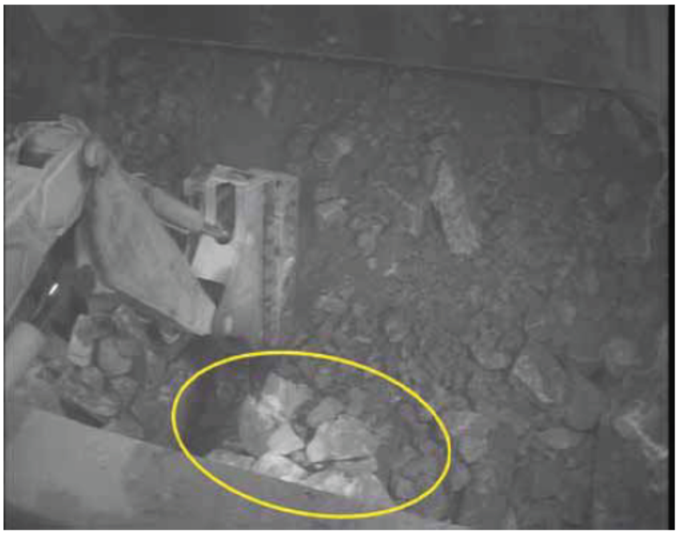


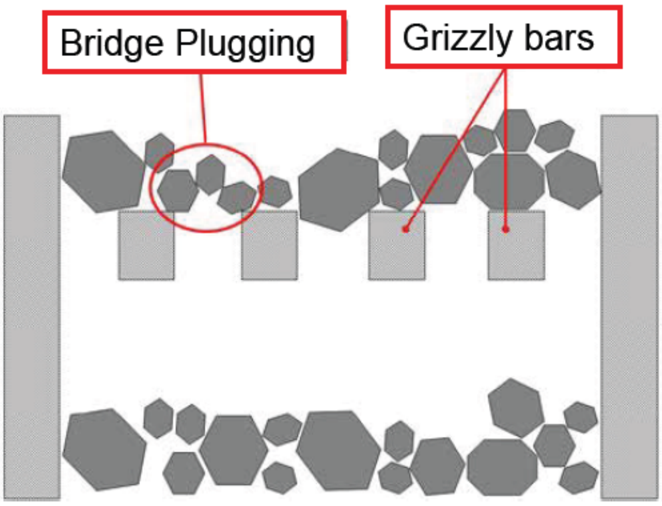
Case2. Plugging with the bar appearing

Locating the Plugged Rock

-Categorization-

The situation can be categorized into 3 categories.

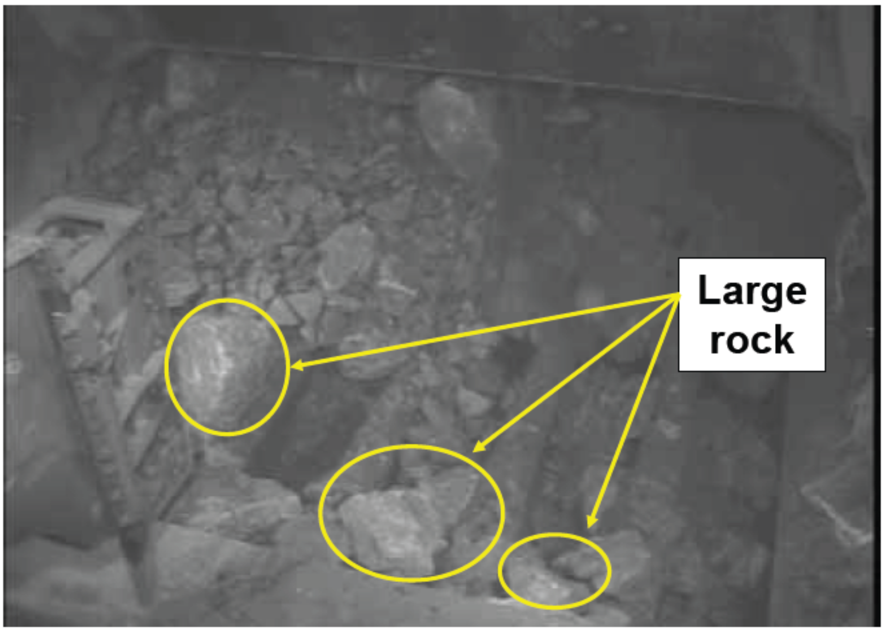




Case3. Bridge Plugging

Locating the Plugged Rock

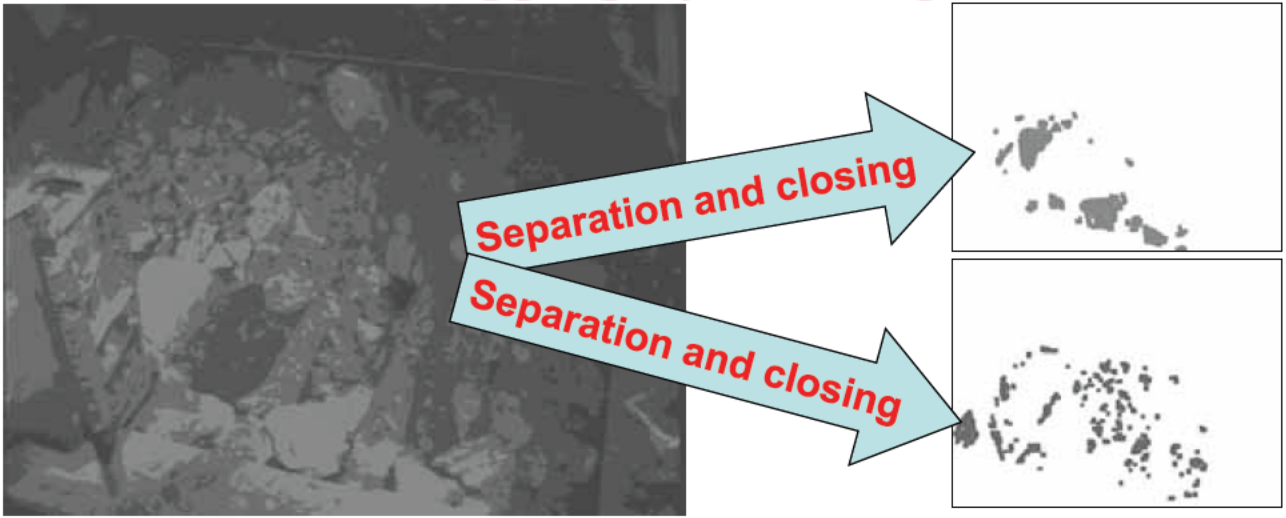
Case1. Plugging by the large rock



1. Surface of the rocks are have bright area.
2. Detect the bright area by image processing.

Locating the Plugged Rock

Case1. Plugging by the large rock



Separation and closing

Separation and closing

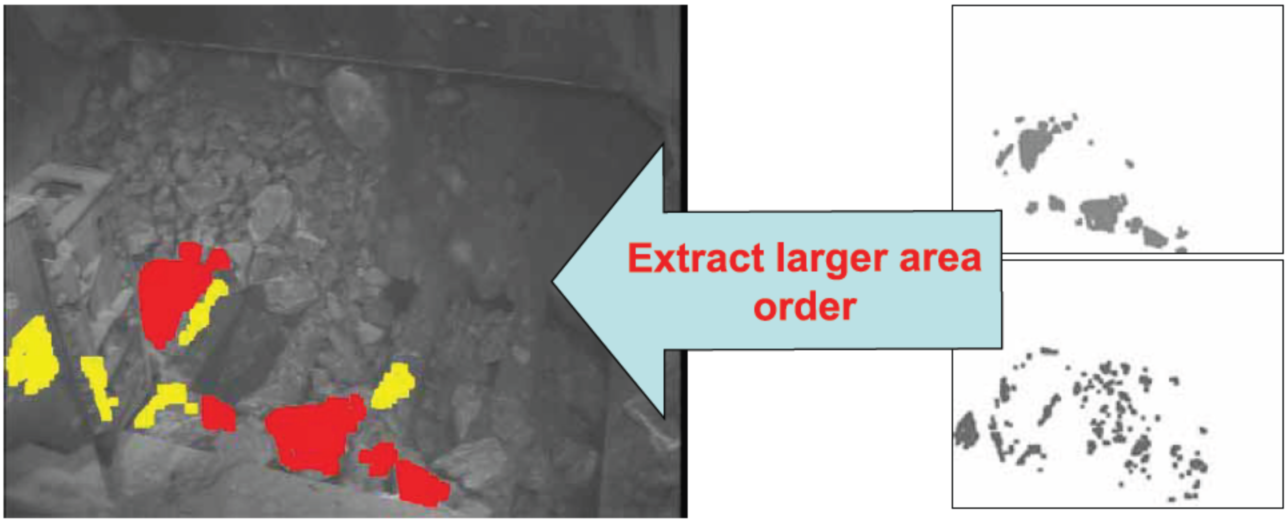
4 gray scale image by k-means method

After separation and closing

- Gray scale transform in to 4 gray scale with k-means method
- The brightest and next brightest blobs are separated and closed

Locating the Plugged Rock

Case1. Plugging by the large rock



Extract larger area order

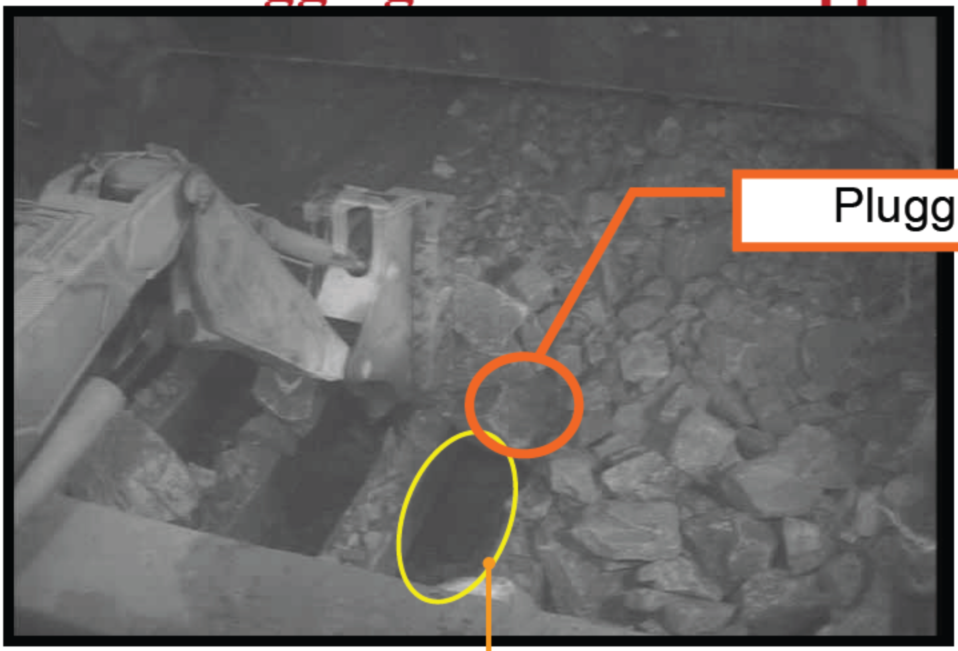
Extracted blobs for the plugging candidates (colored)

After separation and closing

The larger blobs are chosen as candidates of the plugging large rocks.

Locating the Plugged Rock

Case2. Plugging with the bar appearing

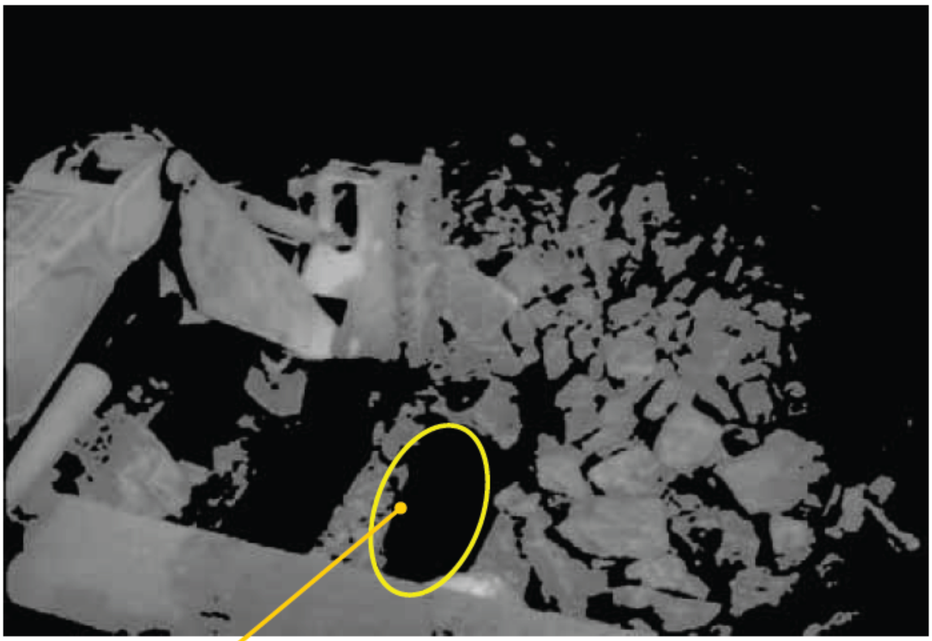


Plugging rock

- Gap between the bars appears before the plugging rock.
- The gap is darker.

Locating the Plugged Rock

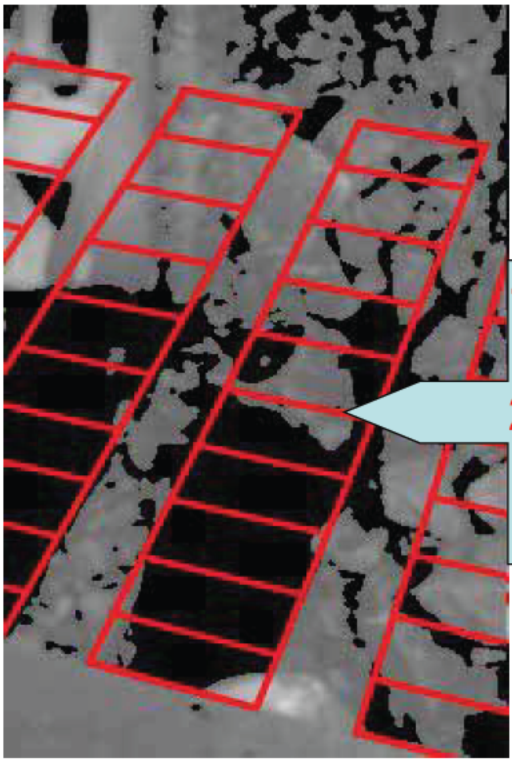
Case2. Plugging by the large rock



Binarization is effective. The gap becomes black.

Locating the Plugged Rock

Case2. Plugging by the large rock

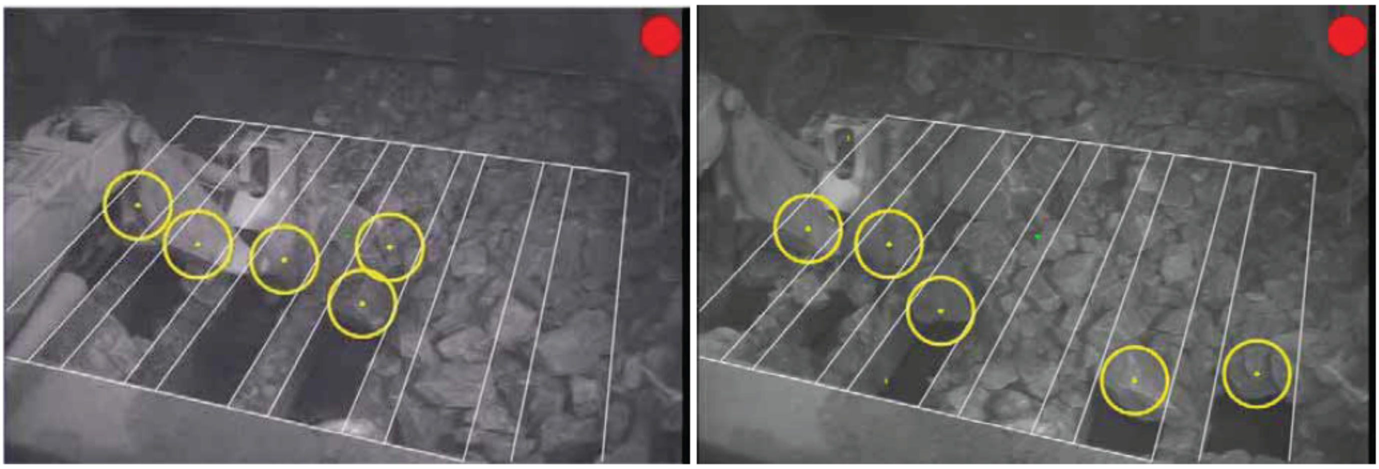


1.Gap between the bars are segmented into several windows.

2. Border of the bright and dark segment window is the candidate of the plugging rock existing.

Locating the Plugged Rock

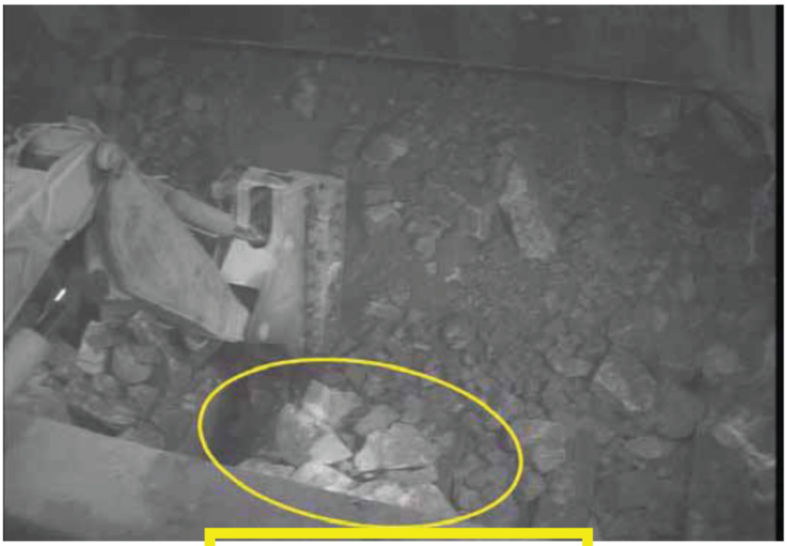
Case2. Plugging by the large rock



Detection examples for the case 2.

Locating the Plugged Rock

Case3. Bridge Plugging

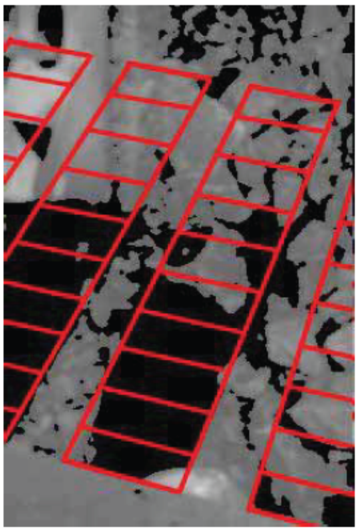


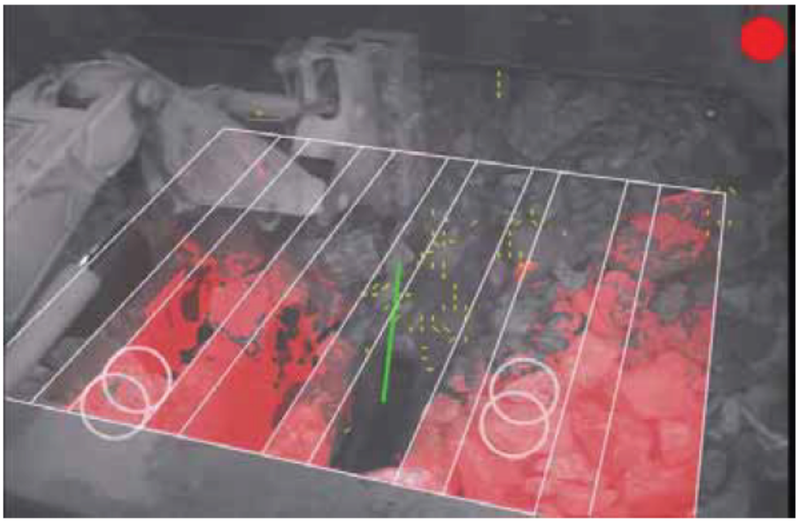
Bridge plugging

- Flow speed is low at the bridge plugging
- Inter frame difference for flow detection is utilized.

Locating the Plugged Rock

Case3. Bridge Plugging





- Investigate every segment window for the plugging with the bars appearing.
- The number of red pixels is superior to black pixels in a segment window when the flow detected.

Rocks Flow Detection and Plugging Location

- Now and Future -

- Now:
  - Detection of flow and stop
  - Location of plugging rock
    - ➢Done on the recorded video images.
- Future
  - Inspection whether the detection and location by the image processing coincides the operator feeling is necessary.
  - Bridge to automatic breaking the plugging rock.

Summary

- Introduction to massive rock breaking operation in the open pit mine
- Introduction of automatic one finger operation for the massive rock
- Introduction of automated detection and location of plugging rocks in the bulldozing chamber

OMG Reston Meeting, 3.19~3.23, March, 2012.

Infrastructure WG Report

Seung-Woog Jung(ETRI)

Infrastructure WG, Robotics DTF

Topic of This Meeting

- Review of the merged submission based on AB comments
- Review of the merged submission on the MARS
  - 15:00~ , Wednesday
- DDC4RTC re-review and voting on MARS
  - 10:00 ~ , Thursday

Infra WG meeting

- Submit a merged document 4-weeks ago
- A lot of good comments from two AB members , Steve and Elisa.
- Monday meeting
  - Review the comments
  - Some minor problems such as formatting, numbering, etc were fixed on Monday meeting.
  - Assigning homework to Infra WG members for major problems
- Tuesday meeting
  - Revised the merged submission based on the homework result

Comment 1-1-1
Document formatting

- Headings without number
  - Numbers are reassigned to all the headings
- Inconsistency between document and TOC.
  - Platform Independent Model has no number
  - TOC has been recreated
- Every pages has “title, version” in its footer
  - “title, version” has been changed to “Dynamic Deployment and Configuration for Robotic Technology Component (DDC4RTC) Specification, draft”

Comment 1-1-2
Document formatting

- The first three elements under Terms and Definitions are redundant and should be removed
  - Removed
- The final section is called “Annex A: Title”
  - “Title” has been changed to “XML Schema and IDL”
- Not all hyperlinks in the Normative References section are active.
  - Annex A was tagged as heading and TOC has been updated

Comment 1-1-3
Document formatting

- The Conformance clause is not a Conformance clause; it is instructions for writing a Conformance clause.
  - Whole sentence of the section has been replaced as the following :

    The DEPL specification which is a basis of this specification defines enables several independent compliance points to enable different vendor implementations or user replacement of implementations. Suggested conformance points are RepositoryManager, TargetManager, NodeManager and ExecutionManager. DDC4RTC follows these conformance points of the DEPL.

Comment 1-1-4
Document formatting

- Some sections start with orphan paragraphs while others start with subheadings called “Introduction”. Be consistent.
  - Subheadings “Introduction” have been removed from 8.2.1, 8.4.1 and 8.5.1
- Section 8.4 has an empty Introduction which should be removed.
  - 8.4.1 has been removed.

Comment 1-2-1
English and grammar

- The first seven paragraphs of section 5 (7) (which is inexplicably named “Introduction” and appears as section 7 in the contents) are almost incomprehensible.
  - rewritten
- The same applies to the two paragraphs in 5.1.1 “Target Environment”. The final paragraph of 5.2 has numerous grammatical errors, errors of spacing, and odd characters. (7.1.1 Target Environment)
  - rewritten

Comment 1-2-2
English and grammar

- Several of the Definitions in Terms and Definitions are too vague.
  - The definitions of SDO and RTC use different words, but I cannot discern any difference in meaning. I do not understand the definition of RT-component profile ? it is unclear what “that” is referring to. The definitions of Environment change, Deployment, Node application, Dynamic global configuration plan, and others are ungrammatical.
  - rewritten
- One you have introduced an acronym (RTC) use it consistently ? there are many references to RT component and RT-component that should all be RTC.
  - Changing RT-component to RTC

Comment 1-2-3
English and grammar

- The section “Symbols” says “there are no special symbols or terms”. This is clearly false since the previous section defined several terms.
  - Changed to “There are no special symbols.”
- The semantics of ApplicationSupervisor, which has no heading numbering at all, stops in the middle of a sentence.
  - Has been revised.

Comment 1-3-1
Architecture

- I understand that the submission consists of a PIM expressed in UML, which extends at least one other existing specification (DEPL). The following things are not clear:
  - The package structure of the model
  - Whether any existing specifications are changed
  - ➢ Sentences "In this specification, the package of …" in the section 8.1 has been revised for the new DDC4RTC specification structure.

Comment 1-3-2
Architecture

- the section "Changes to Adopted OMG Specifications" says that DEPL is extended ? is it changed?
  - Exactly which existing specifications are reused
  - ➢ "This specification just extends the OMG DEPL specification without any changes in the original specification."
  - ➢Sentence in the 6.1 "Changes to Adopted OMG Specification" has been revised.

Comment 1-3-3 Architecture

- A later section talks about extending the “Notification Service Specification” but this is not mentioned in the Normative References
  - Notification Service and its specification URL has been added to “3 Normative Reference”
  - “[NOT] Notification Service Specification, http://www.omg.org/spec/NOT/1.1/“
- 5.2 refers to “D & C” ? what is that? Is that an existing specification or something new?
  - All word “D&C” and the references to D&C have been replaced with DEPL. (Five D&C -> DEPL)
- The PSM specification refers to the UML Profile for CORBA, but that is not mentioned in the Normative References.
  - UML Profile for CORBA has been added to the Normative References section, and all the numbers of other specifications have been updated.

Comment 1-3-4 Architecture

- In the Platform Independent Model section, the text claims that the DDC4RTC package is composed of three subpackages called Dynamic Configuration Planning, Dynamic Configuration Execution and Dynamic Configuration Metadata. This statement does not correspond with the figure, nor with the XMI.
  - The sentence is based on old version of the specification.
  - It has been revised
- An earlier paragraph claimed that the DDC4RTC package consists of RTC Data Model, RTC Execution Model and Dynamic Deployment Model. This doesn’t appear to be true either. In the XMI there is a package called “other”. Overall, the package structure is poorly-documented and unclear.
  - The sentence is based on old version of the specification.
  - 8.1 Overview has been rewritten.

Comment 1-3-5
Architecture

- The diagrams in section 5(8) have the diagram heading “class”. According to the UML specification, this denotes a composite structure diagram. But these are not composite structure diagrams. Some of them appear to be class diagrams representing packages (in which case the header should say package P). But this is not true of ComponentAction under 5.3.3. These diagrams need to be correctly labeled and related to the model.
  - ➢ask to Sakamoto-san by e-mail and then revise the diagram.

Comment 1-3-6
Architecture

- There is a diagram in 5.3.4（）called “RTCDataModel”. Maybe there is supposed to be a package called RTCDataModel? But there is no such package in the XMI.
  - ➢Figure was old version. Updated.

Comment 1-3-7
Architecture

- In 5.3.4 (8.2.6) there are 14 aggregation associations but the text only describes 11.
  - In the 5.4.3 -> 8.2.6 RTComponentActionDescription, attributes lacks some aggregation associations: on_mode_changed, on_startup and on_shutdown

Comment 1-3-8
Architecture

- Section 5.3.6 (changed to 8.2.8) refers to “RTC’s port is same as UML component’s port”. What does this sentence mean? Is it actually referring to Port on UML EncapsulatedClassifier? How are the elements defined in this section supposed to relate to elements described in UML?
  - RTC Port is defined in 5.2 LightweightRTC. It says that *“From [UML]: Ports represent interaction points between a classifier and its environment. The interfaces associated with a port specify the nature of the interactions that may occur over a port. The required interfaces of a port characterize the requests that may be made from the classifier to its environment through this port. The provided interfaces of a port characterize requests to the classifier that its environment may make through this port. “*
  - EcapsulatedClassifier -> BasicComponent::Component, this sentence should refer the figure 5.1 in [RTC]

Comment 1-3-10 Architecture

- Section 5.4 (8.4, 8.4.1 SupervisorFSM) appears to replicate part of UML's state machine metamodel. I am now confused about whether this PIM is supposed to be a metamodel or a model. If you need a state modeler to describe the SupervisorFSM, why not use UML itself? As observed above you seem to be using UML's Ports, so why not use UML's States?
  - ➢We would like to replace SupervisorFSM section with reference to UML' state machine meta model.

Comment 1-3-11 Architecture

- Section 6 refers repeatedly to sections 9.x and 10.x which do not exist. It also talks about "PSM conformance points outlined in chapter 2", which I suppose is intended to refer to the Conformance clause, which is currently numbered 1, and does not contain any conformance statements.
  - ➢Section 9 "Platform Specific Model" has been rewritten.

Comment 1-3-12
Architecture

- The PSM section says “UML classes and interfaces shall be represented as IDL interfaces of the same name”. This seems to me to be a statement that belongs in another standard, not here.
  - Section 9 “Platform Specific Model” has been rewritten

Comment 2-1-0
Cover page

- XMI file should be referenced; there are some blank "Source document" and "Original file(s)" references that should be eliminated.
  - XMI file “mars/2012-02-16 (XMI)” has been added as a source document
  - XML Schema, CORBA IDL files should be provided.

Comment 2-1-1
General, throughout

- There appears to be some sort of formatting error with document footers. Also, the Table of Contents bears little resemblance to the specification itself. Is this the target for the next version of the submission?
  - ➢ Formatting error and document footers has been corrected.

Comment 2-1-2
Preface

- 2nd paragraph needs to be rewritten.
- In the first sentence, replace "changed by physical movement of the robots, and the state transition of the application scenario" with "affected by robot movement and application or scenario state",
  - ➢ The sentence has been replaced.
- and then replace "it is necessary to realize the dynamic deployment of components and run-time re-configuration for the robot applications“ with "it is important to be able to represent and realize dynamic component deployment and run-time re-configuration requirements".
  - ➢ The sentence has been replaced.

Comment 2-1-3
Preface

- Also, change " further propel" to " increase", and "dynamic features in those use cases" with "requirements related to dynamic behaviors".
  - ➢ These words have been replaced.
- 3rd paragraph, first sentence change "configuration feature for the" to "configuration requirements for" and replace "component model" with "component models".
  - ➢ These words have been replaced.

Comment 2-1-4
Scope

- the first paragraph should talk about extending the DEPL rather than only the RTC, shouldn't it?
  - ➢ the first paragraph is replaced with “This specification defines data models and service interfaces of deployment and configuration for RTC (OMG Robotic Technology Component Specification) based dynamic applications as an extension to DEPL (OMG Deployment and Configuration of Component-based Distributed Applications Specification) specification. “

Comment 2-1-5
Conformance

- 2nd paragraph -- change "This specification can be applied to applications which change its system structure during the application life-cycle such as robotic applications." to "This specification is designed to support development of applications whose structure changes dynamically at run-time, such as robotic applications." ... or something like that if this is too limiting. I'm not sure that life-cycle is the right term here, and it's used again in a similar way in the second sentence. Rather, I think what is meant has to do with application state and evolution over time. Perhaps this should be restated a bit more clearly, as it is an important point.
  - ➢ The sentence has been replaced.
- Conformance needs to be specified. This is a critical omission, and needs to be addressed as soon as possible.
  - ➢ Conformance has been specified.
- some are blank at the start of this section.
  - ➢ Re-formatted.

Comment 2-1-6
Terms and definitions

- Also, I don't understand the definition of Domain application. Clearly this must be a contextual reference within robotics, but perhaps the definition can be expanded or rephrased a bit. I get what is meant by a multi-node application, but what does that have to do with the domain in which it is deployed? A multi-node application may or may not provide full coverage for some capability in a particular domain, in other words, but what makes it unique to that domain (where domain means a particular area of interest, or well-defined context)? This usage seems prevalent across a number of definitions, so perhaps this is jargon in robotics, but it is misleading in my view.
  - ➢ This section has been revised.

Comment 2-1-7 Introduction

- End of the first paragraph, change "such as real-time system, the system is changed dynamically are defined." to "in dynamic real-time systems is needed." 2nd paragraph needs to be rewritten, and I'm not sure exactly what to do there.
- 3rd paragraph -- change "On the other hand, in the robot system, robot and devices, in order to move in terms of network topology or physical, also dynamically change the placement of components attached to them." to "In a robotic system including robots, sensors, and/or other devices, there is a requirement for such elements to move, physically and in terms of network topology, and potentially to dynamically change internal state and component configuration."
- Change "Furthermore, it is switched by an event external scenario of robot operation, system configuration is changed dynamically due to the often it happens." to "In addition to internally triggered changes in state, location, and configuration, external environmental factors can impact operations dynamically."
- Change "That is, even during system operation as well as at the start of system operation, including redeployment and configuration of components, connections and settings between the components is carried out." to "That is, component configuration, connections, and settings established at the start of operation can be changed at run time in significant ways, either due to internal state change or external situational change."
- Change "By applying an extension of dynamic features to support these standards DEPL, to achieve the deployment and configuration suitable for the robot using the dynamic characteristics of the RT component becomes possible." to "Support for these kinds of dynamic reconfiguration and state change requirements using RT components is possible by extending the DEPL standard as described herein."
- 4th paragraph -- delete the first sentence and add the second sentence to the end of the third paragraph.
  - ➢ Introduction section has been rewritten.

Comment 2-1-8 Section 5.1 (7.1)

- Dynamic Deployment and Configuration -- the first three paragraphs need a careful rewrite, which I can help with offline, as needed. The remainder of the section is quite good, however. Last paragraph, change "stat einformation" to "state information".
  - ➢ Has been modified.
- 5.1.1 Target Environment -- needs to be re-written, and maybe should not be a separate section unless there is a 5.1.2 (none at the moment). Again, I can help with the rewrite offline, but it definitely should be made clearer.
  - ➢ It has be rewritten.

Comment 2-1-9
Section 5.2 (8.1)

- This is fairly well written. There is a parenthetical at the end of the caption under Figure 8.1, which should be 5.1? in Japanese -- please revise or delete :).
  - Deleted.
- Also, the last sentence of the final paragraph in 5.2, "In this case refered class name is described with its name space of package." should probably be "In this case, the referenced class is includes the appropriate namespace information." or something like that.
  - Replaced.

Comment 2-1-10
Section 5.3 Component Data Model

- not sure why the ComponentDataModel namespace is referenced within the diagram; isn't that the namespace of this model?
  - ComponentDataModel namespace is DEPL package. The diagram should be revised.
- all figures in this section need numbers and captions
  - Numbers and Captions have been added to all the figures.
- 5.3.2 attribute definitions need a bit of editing, and highlights should be removed
  - Done
- ExecutionType and ActivityType are not numbered (formatting issue), and ActivityType is not referenced in any diagram
  - Done
- 5.3.4 RTCImplementationDescription Semantics paragraph on page 24 of 43 (pdf pages, not as they are numbered) needs a bit of editing
  - Semantics paragraph of 8.2.6 should be reviewed
- 5.3.6 PortInterfaceinstanceType heading moved to 5.3.7, where it is shown in the diagram? Capitalization should be corrected to PortInterfaceInstanceType, as it is in the text of the description?
  - Section "8.2.9RTCPortInterfaceDescription" and "PortInterfaceinstanceType" are replaced. Type corrected.
- 5.3.7 Minor editing needs to be done on the Description; RTCSubcomponentPortEndPoint belongs with which diagram? Should this be in a separate section?
  - A figure has been added. It should be rewritten.

Comment 2-1-11
Section 5.4 Execution Data Model

- Introduction has no text?
  - Introduction removed.
- Subelements should be numbered? The descriptions are fairly thin, although I don't necessarily have an issue with this. This provides yet another finite state machine model, though -- do we have others in OMG specifications, including UML, that could be used or at least extended for this?
  - Using UML state machine meta-model should be discussed.
- Execution Management Model should be in a separate section? It is referred to in the text as Event Management Model, so whichever it is, there should be consistency here again, all these figures should be numbered and have captions associated with them
  - All the figures have been numbered.

Comment 2-1-12
Section 6 (9)

- as mentioned in my summary notes, this chapter refers to sections 9 and 10 that are not part of the specification, so it is possible that the numbering is simply "off" or that it is referring to a different specification. This should be fixed, one way or another. Also, the transformation rules are stated in text here -- is that sufficient? It seems a bit lightweight to me, or perhaps it is sufficient, but I would like to see some sort of break out of the content of the Generic Transformation Rules paragraph, and possibly a flow diagramof some sort to go with it.
  - This section should be rewritten carefully.

Comment 2-1-13
Annex A

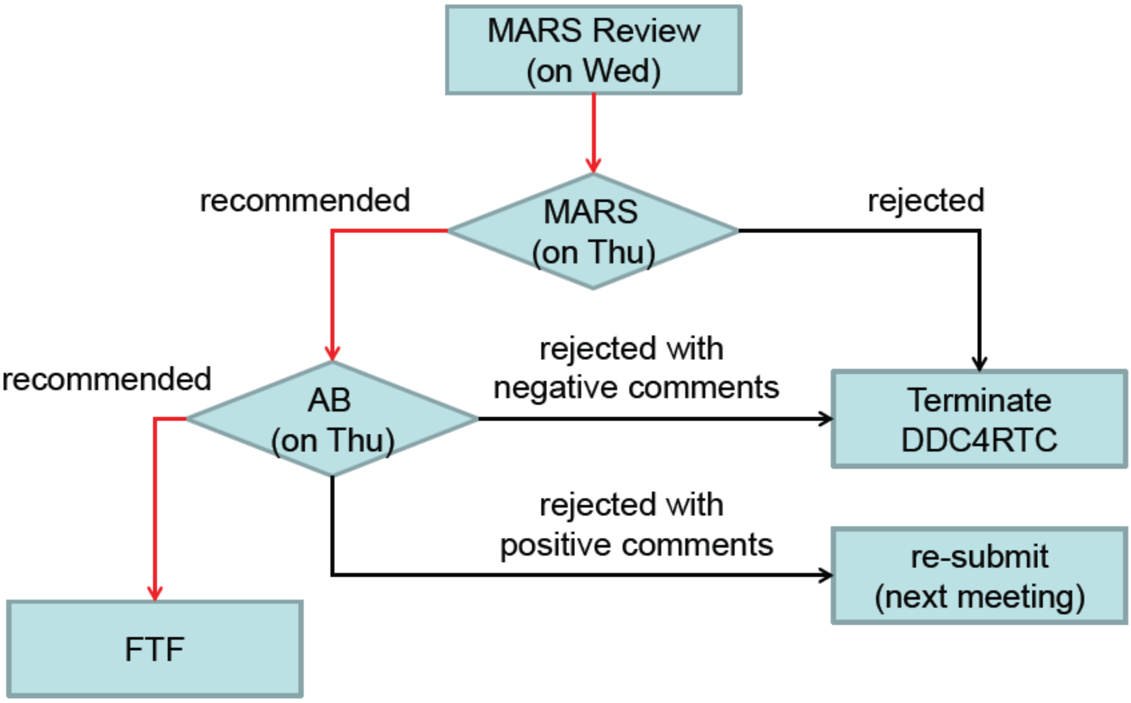
- Annex A does not match the XMI file -- one telling example is that at the top of the diagram, where encoding is "ISO-8859-1" in the document, it is "SHIFT-JIS" in the XMI file, and a significant number of EA-specific statements are not present in the XMI in the document, so clearly they are different. They need to be reconciled. Also, it appears that there are two schema in the document, but only one XMI file -- have they been combined?
  - ➢ XMI should be revised.

Comment 2-1-14
XMI file

- There is quite a bit of EA-specific stuff in the XMI file, all of which needs to be removed. I inspected the XMI with a couple of different browsers, and there seems to be some EA diagram-specific material in particular that should be removed. You might try using a more recent version of EA, and exporting the XMI from that, in order to eliminate it. I have not yet had time to load this in EA and look at it in more detail, but will do so later in the week and send further comments, if any.
- I hope this is helpful, and look forward to talking further with you about it. Please feel free to contact me off-list, although my mother is quite ill, and so my ability to spend a lot of time on this is limited at the moment. Again, my general comments were in the earlier email and I wish you the best in moving it forward. I do understand the need for the capability, and appreciate the work that has gone into this.

➢XMI should be revised.

Future Plan



MARS Review (on Wed)
recommended
MARS (on Thu)
rejected
recommended
AB (on Thu)
rejected with negative comments
Terminate DDC4RTC
rejected with positive comments
re-submit (next meeting)
FTF

Robotic Functional Service WG
WG Report

WG Co-Chair: Toshio Hori
2012/03/20

WG activities before this meeting

- At Seoul private Meeting (2012/02)
  - Discussed issues for RoIS Framework
    - Attendants: Chi(ETRI), Cho(ETRI), Doi(Toshiba), Kamei(ATR), Nishio(ATR), Sato(Denso), Tsubouchi(Univ. of Tsukuba), Hori(AIST)
  - 26 issues were raised and discussed.
    - 15 issues were resolved.
    - 11 issues were discussed. They were resolved partially but we couldn't get clear resolutions.
- Comments due passed (2012/02/20)
  - One new issue was raised by the deadline by Hori.

WG activities during this meeting

- Monday
  - Issues and resolutions were listed up for voting.
  - There were comments on resolutions to 2 (resolved) issues. We need more discussions to reach an agreement.
    - Currently, 13 resolved issues, 2 issues require more discussions, 12 unresolved issues.

Schedule after this meeting (tentative)

- Discuss unresolved issues.
- Make a final draft of the RoIS specification.
- Present FTF report by the deadline (21st May)

- We may have another private meeting in Seoul in May (one week before the FTF report deadline).

- Deadlines (indicated in the charter):
  - ~~Comments Due: 20th February, 2012~~ PASSED
  - Report Due Date: 21st May, 2012
  - Report Deadline: 29th June, 2012

Issue #	Title	Resolution
	Description	
16560	Command Result Message definitions	
	In 7.5.2 there is no profile for the result parameters related with 7.4.3.2	
16893	Definition of HRI Engine is ambiguous	The definition is changed as follows: An object that manages HRI Components. It mediates Human-Robot Interaction functions of the HRI Components to Service Application(s).
	In Section 4, definition of HRI Engine is ambiguous	
16894	Order of HRI Engine and Service Application should be changed	Their order is changed because the term "HRI Engine" refers to "Service Application." [Comment] The terms "Localization" and "Identification" refer to the term "Identifier" which is defined after them so "Identifier" should be defined before them. But, in my opinion, the terms "Detection," "Localization" and "Identification" may be treated as a set of HRI functions so I recommend a new order in the table as follows: "Identifier," "Detection," "Localization," and "Identification." [Comment2] The order of terms could be alphabetical order. Keywords could be underlined (and be cross referenced if possible).
	In Section 4, Order of HRI Engine and Service Application should be changed	
16895	Relationship with RTC and ROS	
	Relationship with ROS, RTC should be clarified	
16923	Parameter definitions for Error Message	
	In 7.4.3 Message Data, there is no definition of Error Message for get_error_detail method, and there is no profile for the result parameters for this message in 7.5.2 7.5.3.	
16924	Occurrence of sub component for Component Profile	After the discussions at the Santa Clara meeting, we concluded that the occurrence of sub_component in HRI Component Profile should be changed as 1.
	In Table 7.29 Component Profile, occurrence of sub_component is N to avoid double definition. For example, if the each sub component include RoIS_Common, the main component may include the two same common messages defined in RoIS_Common. Therefore, the occurence should be "1".	
16925	Parameter in HRI Engine Profile	This comment makes sense so we remove the definition of "parameter" from the profile.
	In Table 7.30 HRI_Engine_Profile, parameters of HRI engine is defined in this profile. However, these parameter should be defined in the Component Profile of System Information Component for this Engine. Therefore, "parameter" definitions in HRI engine profile should be removed.	
16926	RoIS Common in System Information component	This is a copy-paste mistake so RoIS_Common should be removed.
	In Figure 17, RoIS_Common should be removed because System Information component does not include RoIS Common messages.	
16927	Parameters for System Information component	A system may operate several robots in an HRI Engine when the engine has sub HRI Engines or it controls multiple robots. In this case, each robot position should be distinguished by robot's ID (i.e. robot ref). So the ordered lists of robot ref and position are returned as result parameters.
	In Table 7.34 System Information, robot ref(List<RoIS_Identifier>) is required for robot_position, same as person position data in "Person localization" In addition, position_data should be List<Data>.	

Issue #	Title Description	Resolution
16928	Parameters for Localization component In Table 7.36 person_localization(and other localization components), parameters of sensing-cycle and/or resolution (minimum position difference which sensor can distinguish) are required. These parameters should be added for (optional) get_parameter. In addition, Detection-cycle (multiple of sensing-cycle) and Detection-difference (larger than resolution) may be set by set_parameter.	This is a matter of Robotic Localization Service (RLS). [Comment] We should study RLS to know how parameters/arguments are passed to/from the RLS module, then add Command/Query Messages that accept the parameters/arguments appropriately to the table.
16929	Parameters for Speech Synthesis component In Table 7.44 speech synthesis, data type of the argument and result parameter of "character" should be RoIS_Identifier.	"Character" in this context indicates male, female, adult, child, etc. and the available character types depend on a speech synthesis component. To distinguish and designate a character type in Command Method (set_parameter) and Query Method
16930	Additional Normative References In 3.1 Normative References, ISO19143 (Geographic information – Filter encoding) should be added in the list. This reference is referred for QueryExpression.	The reference was missed when submitting a draft so it should be added.
16931	Notification of Profile change It is required to consider methods about notification when the engine or component profile changed. "receive_error" method is usable for this purpose. PROFILE_CHANGED(tentative name) should be added in the enumeration of "Error Type" (Table 7.7). However, this status is not an error, so "receive_error" and "Error Type" should be renamed.	
16932	Wrong Expression P.23 line 3, "each type of error" should be corrected to "each error."	This is a grammatical error so it should be corrected.
16933	Condition for identifying message Condition of get_error_detail (Table 7.2), get_command_result (Table 7.3), and get_event_detail (Table 7.5) can be omitted because these method can identify their message by message_id (such as command_id, event_id and error_id).	
17133	Atomic allocation of multiple HRI Components required Methods "search," "bind" and "bind_any" may be extended for allocating multiple components at once avoiding race condition.	
17134	Dependency among HRI Components should be defined Dependencies (Requisite conditions) among HRI Components should be defined in HRI Profile.	
17135	Miss speling in Table 7.2 In Table 7.2 (p.31), the word "Enghien" should be "Engine." (in the "connect" row)	TYPO. Should be corrected.
17136	Unifying Notation of Terms "HRI Engine," "HRI Components" and other terms defined in Section 4 should be unified as in the definition table.	For example, "engine" and "component" are general words while the terms "HRI Engine" and "HRI Component" are specific to this document. To distinguish them, the notation of specific terms will be unified as "HRI Engine," "HRI Component," etc. as in Section 4.
17137	Complex definition of "Command Unit List" Implementing "Command Unit List" is very difficult. It should be simplified.	
17138	Ambiguous definition of "Condition" "Condition" is used in many interfaces and method, but it is difficult to implement without example.	
17139	General System Message Class should be defined The structure of message-related classes should be organized concerning issues #16560, #16923 and #16931.	

Issue #	Title Description	Resolution
17140	Table number missing in 7.6	TYPO. Should be added.
	In p. 46, line 3, Table number (7.31) is missing.	
17141	″Supported by″ in Acknowledgements should be modified	As these organizations supported drafting a document, I'd like to add them to the acknowledgment list.
	Miss speling in ATR (missing 'd'). Several organization should be added such as Korean Robot Association, Future Robot Co., Ltd., University of Tokyo and Hitachi.	
17142	Copyright holders should be modified	These organizations were added by misunderstanding. So I propose removing them.
	ATR and AIST should be removed from copyright holders.	
17143	Two types of parameters to be distinguished	
	There are two types of parameters to be distinguished. For example, ″language″ and ″character″ in speech synthesis are stable ones, but on the other hand, ″speech_text″ and ″volume″ are instant ones. Engine or Component should re-transmit or interpret them correctly.	
17145	Schematic diagram of RoIS Framework (Figure 5, in Section 7.2) requires an update	
	Interfaces between the Service Application and HRI Engine had been changed when we submitted the revised submission but the diagram had not updated upon submission.	

🟧	Resolved issue
🟩	Resolved but requires discussions
⬜	Unresolved issue

SIMPAR-2012

http://www.simpar.org/

The 3rd International Conference on SIMULATION, MODELING, and PROGRAMMING for AUTONOMOUS ROBOTS

November 5-8 2012
Tsukuba
in JAPAN

Scope

Novel robotics applications driven by research, industry and society call for the development of systems of ever increasing complexity: systems with sliding autonomy; humanoid robots; distributed robots; mobile sensor networks, and so on. But unfortunately, steady improvements in robot hardware have not been matched by corresponding advancements in robot software. Besides fundamental open problems still waiting for sound answers, the development of new robotics applications still suffers from the lack of widely used tools, libraries, and algorithms ready to be incorporated into new projects. Simulation environments are playing a main role in reducing development time and cost of large scale systems. But their use is still regarded by many with skepticism. Seamless migration of code from general purpose simulators to real world systems is still a rare circumstance, due to the complexity of robot, world, sensors, and actuators modeling.

These challenges drive the quest for next generation of methodologies and tools for robot development. The objective of the International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR) is to offer a unique forum for these topics and to bring together researchers from academia and industry to identify and solve the key issues necessary to ease the development of increasingly complex robot software, and to boost a smooth shifting of results from simulated to real applications.

Conference Committees

General Chair: Itsuki Noda (AIST, Japan)

Steering Committee:

- Tamio Arai (University of Tokyo, Japan)
- Herman Bruyninckx (Katholieke Universiteit Leuven, Belgium)
- Xiaoping Chen (University of Science and Technology of China)
- Maria Gini (University of Minnesota, USA)
- Enrico Pagello, Founding Chair (University of Padua, Italy)
- Lynne Parker (University of Tennessee, USA)
- Oskar von Stryk (University of Darmstadt, Germany)

Program co-Chairs

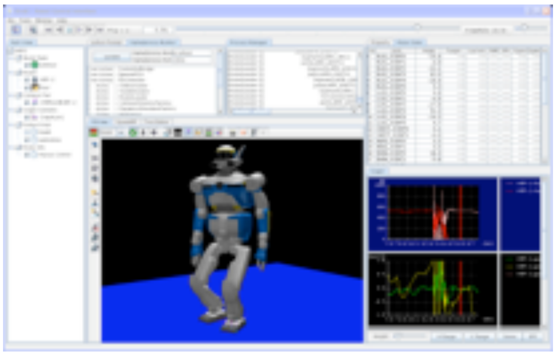
- EU: Davide Brugali (University of Bergamo, Italy)
- US: James Kuffner (CMU, USA)
- AP: Noriaki Ando (AIST, Japan)

Topics of interest include, but are not limited to:

- Robot simulation and mathematical modeling of robots
- Reliability, scalability and validation of robot simulation
- Simulated sensors and actuators
- Offline simulation of robot design
- Online simulation with realtime constraints
- Simulation with software/hardware in the loop
- Modeling framework for robots and environments
- Robotic service by ubiquitous sensor network
- Interaction between sensor networks and robots
- Communication infrastructures in distributed robotics and sensors
- Human robot interaction and collaboration
- Multirobot systems
- Software platform and middleware for robotics
- Testing and validation of robot software
- Standardization for robotic services

Important Dates (tentative)

- Deadline for submission of papers:
  May 15, 2012
- Proposal for Tutorials/workshops:
  April 15, 2012
- Notification:
  July 15, 2012
- Submission of final camera-ready-papers:
  August 15, 2012



Robotics-DTF Plenary Meeting
Wrap-up Session

March 20, 2012

Reston, VA, USA

Hyatt Regency Reston

Document Number

mars/2012-02-15 Dynamic Deployment and Configuration for RTC (DDC4RTC) submission
mars/2012-02-16 DDC4RTC XMI
mars/2012-02-18 DDC4RTC Inventory
mars/2012-03-25 DDC4RTC Specification Presentation (Noriaki Ando)
mars/2012-03-26 DDC4RTC Convenient document without change bar
mars/2012-03-27 DDC4RTC Convenient Document with change bar
mars/2012-03-28 DDC4RTC Errata
mars/2012-03-29 DDC4RTC XMI

robotics/2012-03-01 Final Agenda (Tetsuo Kotoku)
robotics/2012-03-02 Santa Clara Meeting Minutes [approved] (Seung-woog Jung and Koji Kamei)
robotics/2012-03-03 Opening Presentation (Tetsuo Kotoku)
robotics/2012-03-04 Roadmap for Robotics Activities (Tetsuo Kotoku)
robotics/2012-03-05 A Trial Approach for Automation in Open Cut Mine (Takashi Tsubouchi)

Document Number

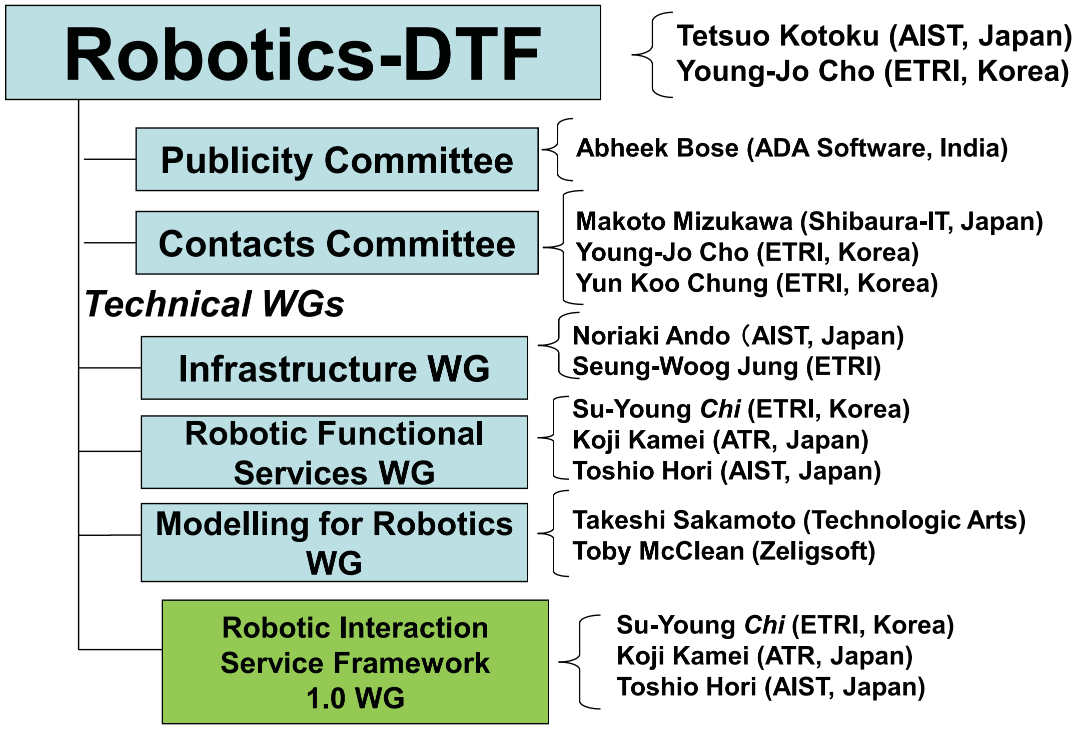
robotics/2012-03-06 Infrastructure WG Report (Seung-Woog Jung)
robotics/2012-03-07 Robotic Functional Services WG Report (Toshio Hori)
robotics/2012-03-08 RoIS-FTF Issues and Resolutions (Toshio Hori)
robotics/2012-03-09 Call for Paper: 3rd International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR2012)
robotics/2012-03-10 Wrap-up Presentation (Tetsuo Kotoku)
robotics/2012-03-11 DDC4RTC Specification Presentation [mars/2012-03-25] (Noriaki Ando)
robotics/2012-03-12 Next Meeting Preliminary Agenda - DRAFT (Tetsuo Kotoku)
robotics/2012-03-13 DTC Report Presentation (Noriaki Ando)
robotics/2012-03-14 Reston Meeting Minutes - DRAFT (Geoffrey Biggs and Seung-woog Jung)

Call for volunteer

- Robotics-DTF Co-Chair

  => Postpone voting one more meeting

Organization (from March. 20th, 2012)



Robotics-DTF
- Tetsuo Kotoku (AIST, Japan)
- Young-Jo Cho (ETRI, Korea)

Publicity Committee
- Abheek Bose (ADA Software, India)

Contacts Committee
- Makoto Mizukawa (Shibaura-IT, Japan)
- Young-Jo Cho (ETRI, Korea)
- Yun Koo Chung (ETRI, Korea)

Technical WGs

Infrastructure WG
- Noriaki Ando（AIST, Japan)
- Seung-Woog Jung (ETRI)

Robotic Functional Services WG
- Su-Young *Chi* (ETRI, Korea)
- Koji Kamei (ATR, Japan)
- Toshio Hori (AIST, Japan)

Modelling for Robotics WG
- Takeshi Sakamoto (Technologic Arts)
- Toby McClean (Zeligsoft)

Robotic Interaction Service Framework 1.0 WG
- Su-Young *Chi* (ETRI, Korea)
- Koji Kamei (ATR, Japan)
- Toshio Hori (AIST, Japan)

Next Meeting Agenda (Long version)

June 18-22 (Cambridge, MA, USA)

Monday:

DDC4RTC revised submission review, vote-to-vote, voting (am)
WG activity (pm)

Tuesday:

WG activity (am)
Robotics-DTF Plenary Meeting (pm)
- **Guest and Member Presentation**
- **Contact reports**

Wednesday:

WG activity follow-up

Thursday:

RoIS Final Report (pm)
WG activity follow-up [if necessary]

Next Meeting Agenda (short version)

June 18-22 (Cambridge, MA, USA)

Tuesday:

WG activity [Parallel WG Session] (am)
Robotics-DTF Plenary Meeting (pm)
- **Guest and Member Presentation**
- **Contact reports**

Wednesday:

WG activity follow-up [if necessary]

Thursday:

RoIS Final Report (pm)

Plenary Attendee (6 participants)

- Geoffrey Biggs (AIST)
- Noriaki Ando (AIST)
- Seung-Woog Jung (ETRI)
- Takashi Tsubouchi (Univ. of Tsukuba)
- Tetsuo Kotoku (AIST)
- Toshio Hori (JARA/AIST)

robotics/2012-03-11

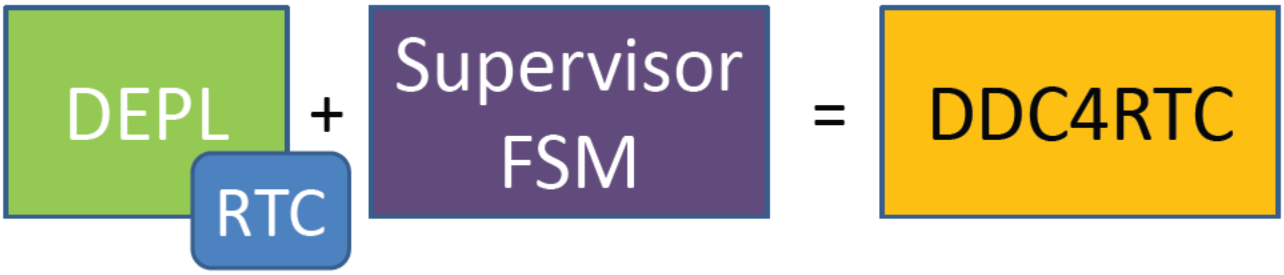
DDC4RTC

Infra. WG, Robotics DTF

mars/12-03-25

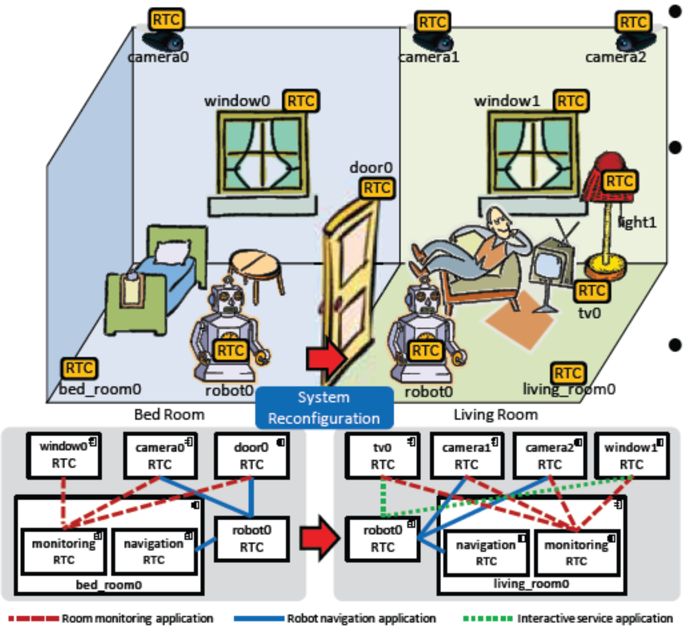
DDC4RTC Specification Overview

- RFP: Minneapolis meeting, Jun. 2010
  - mars/10-06-16 (Deployment and Dynamic Configuration (DDC) of Robotic Technology Components (DDC4RTC) RFP
- Submitters: ETRI, AIST
- Initial Submissions: Santa Clara Meeting Dec. 2010



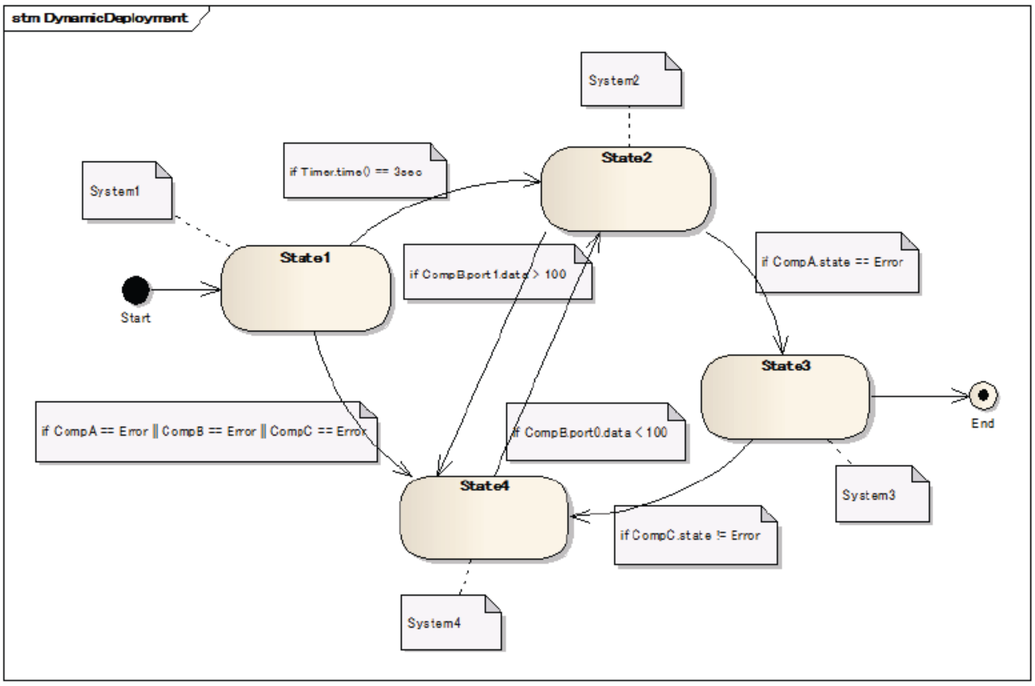
- DEPL: Deployment and Configuration of Component-based Distributed Applications Specification
- RTC: Robotic Technology Component specification

Motivation

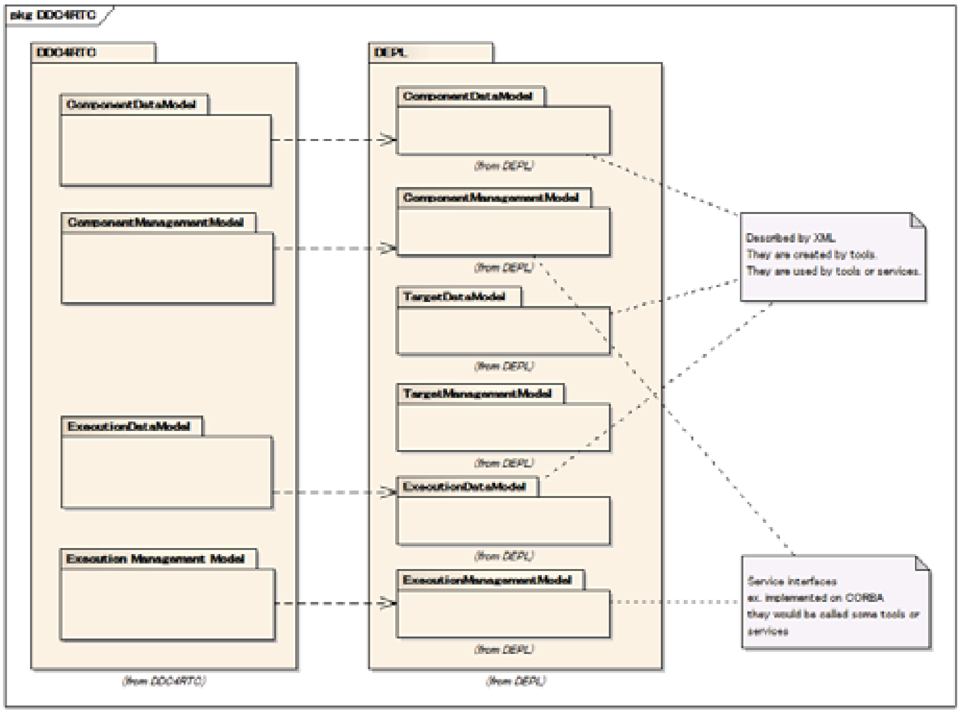


- Many RTCs are distributed spatially
- Systems would be constructed as RTCs aggregation
- System structure should be changed according to the environmental changes in run-time

SupervisorFSM



DEPL and DDC4RTC



pkg DDC4RTC

DDC4RTC

ComponentDataModel

ComponentManagementModel

ExecutionDataModel

Execution Management Model

(from DDC4RTC)

DEPL

ComponentDataModel

(from DEPL)

ComponentManagementModel

(from DEPL)

TargetDataModel

(from DEPL)

TargetManagementModel

(from DEPL)

ExecutionDataModel

(from DEPL)

ExecutionManagementModel

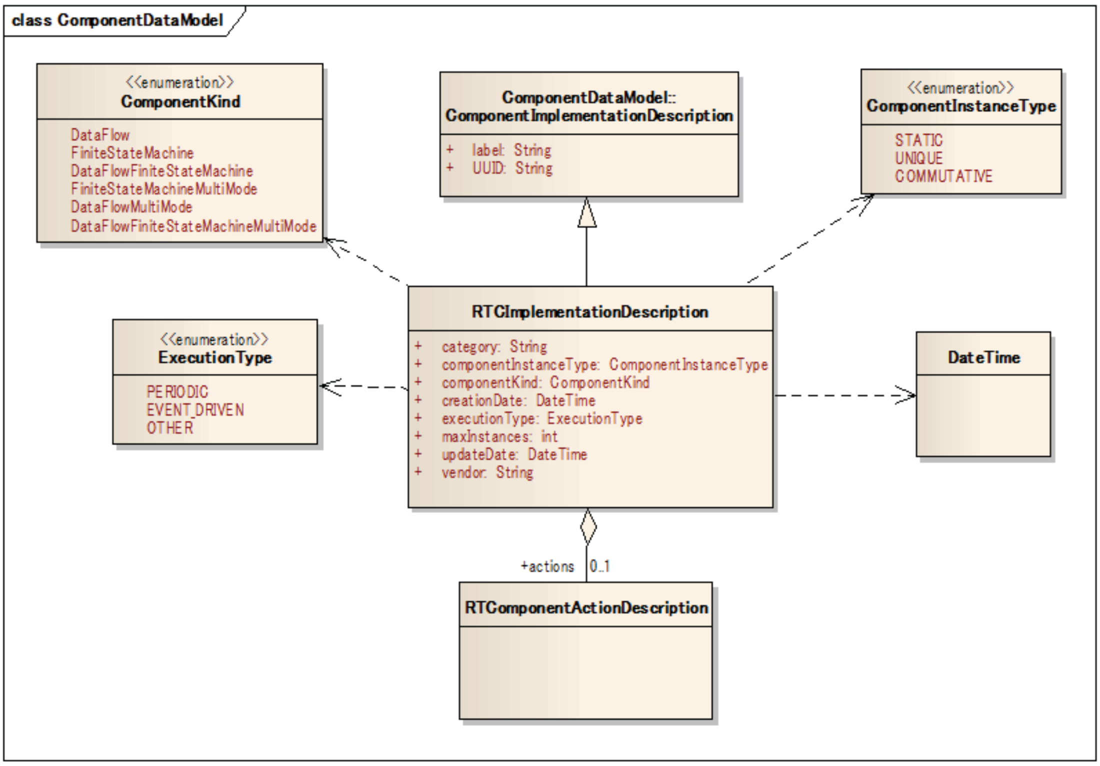
(from DEPL)

(from DEPL)

Described by XML.
They are created by tools.
They are used by tools or services.

Service interfaces
ex. implemented on CORBA
they would be called some tools or services

Component Data Model



class ComponentDataModel

<<enumeration>>
ComponentKind

DataFlow
FiniteStateMachine
DataFlowFiniteStateMachine
FiniteStateMachineMultiMode
DataFlowMultiMode
DataFlowFiniteStateMachineMultiMode

ComponentDataModel::
ComponentImplementationDescription

+ label: String
+ UUID: String

<<enumeration>>
ComponentInstanceType

STATIC
UNIQUE
COMMUTATIVE

<<enumeration>>
ExecutionType

PERIODIC
EVENT_DRIVEN
OTHER

RTCImplementationDescription

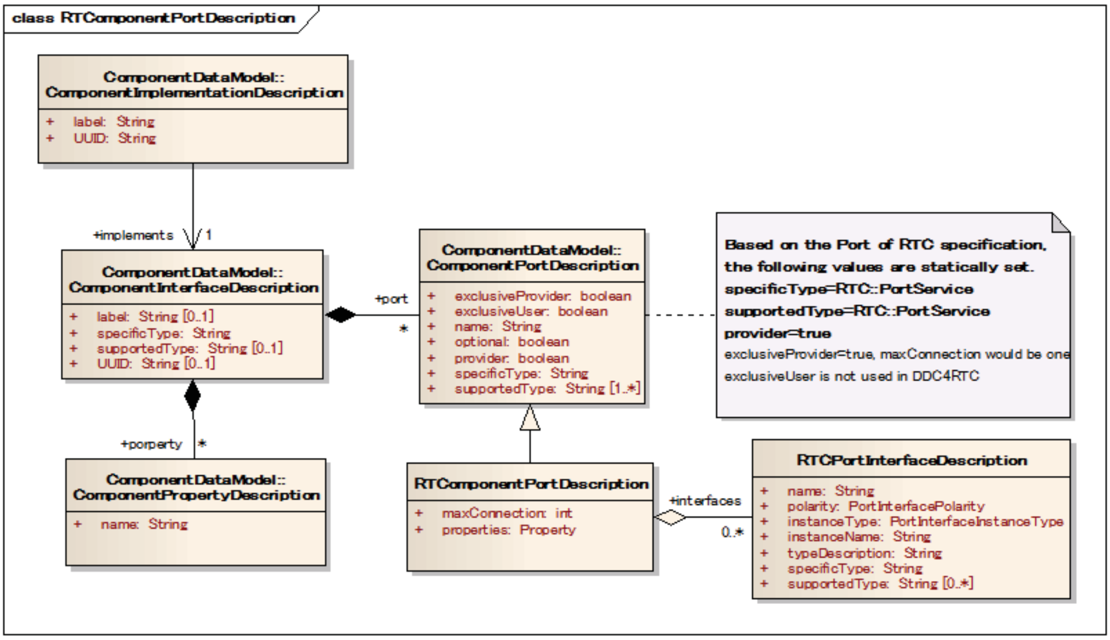
+ category: String
+ componentInstanceType: ComponentInstanceType
+ componentKind: ComponentKind
+ creationDate: DateTime
+ executionType: ExecutionType
+ maxInstances: int
+ updateDate: DateTime
+ vendor: String

DateTime

+actions 0..1

RTComponentActionDescription

RTComponentPortDescription



class RTComponentPortDescription

ComponentDataModel::ComponentImplementationDescription

+ label: String
+ UUID: String

ComponentDataModel::ComponentInterfaceDescription

+ label: String [0..1]
+ specificType: String
+ supportedType: String [0..1]
+ UUID: String [0..1]

ComponentDataModel::ComponentPropertyDescription

+ name: String

ComponentDataModel::ComponentPortDescription

+ exclusiveProvider: boolean
+ exclusiveUser: boolean
+ name: String
+ optional: boolean
+ provider: boolean
+ specificType: String
+ supportedType: String [1..*]

RTComponentPortDescription

+ maxConnection: int
+ properties: Property

RTCPortInterfaceDescription

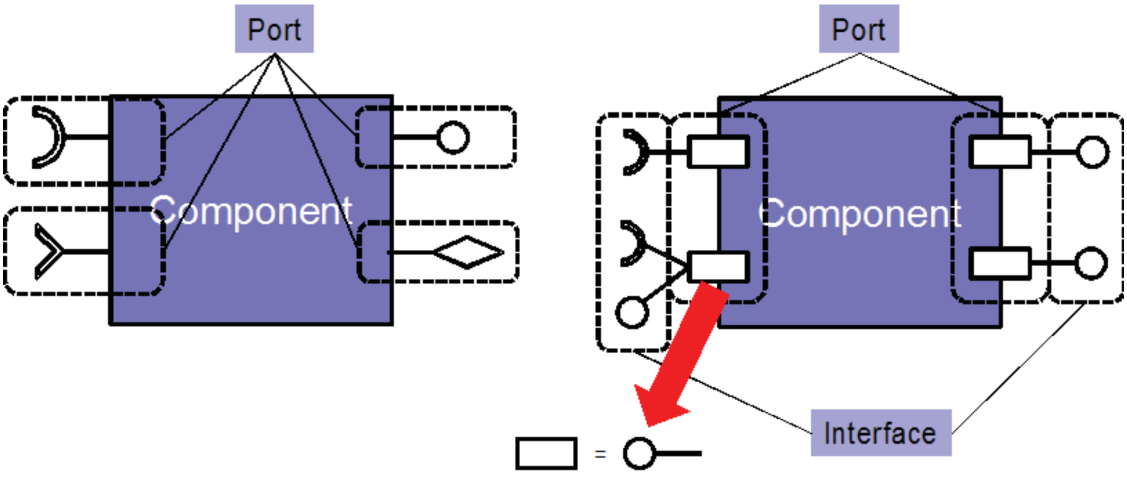
+ name: String
+ polarity: PortInterfacePolarity
+ instanceType: PortInterfaceInstanceType
+ instanceName: String
+ typeDescription: String
+ specificType: String
+ supportedType: String [0..*]

Based on the Port of RTC specification, the following values are statically set.
specificType=RTC::PortService
supportedType=RTC::PortService
provider=true
exclusiveProvider=true, maxConnection would be one
exclusiveUser is not used in DDC4RTC

Port in DEPL and Port in RTC

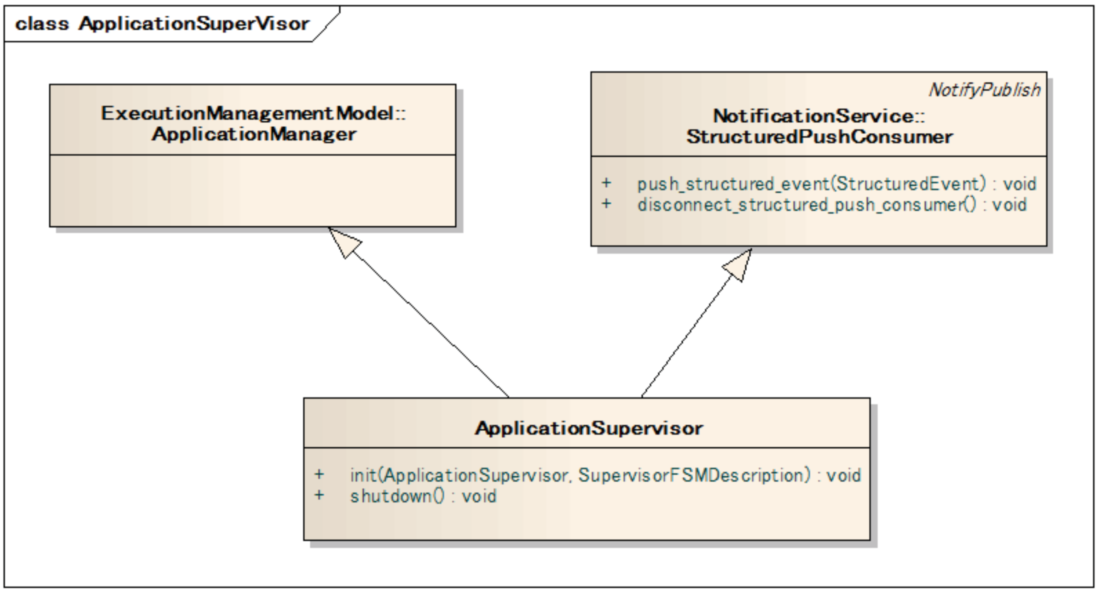
Port and Component in DEPL

Port and Component in RTC

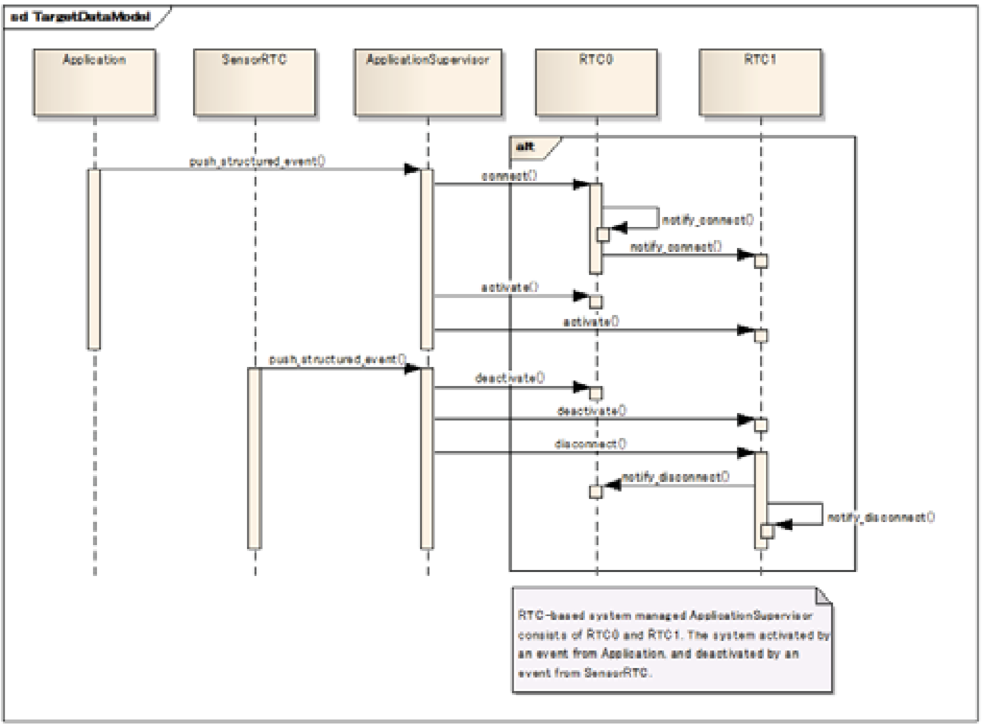


Port itself is a service (RTC::PortService)

ApplicationSupervisor


class ApplicationSuperVisor
ExecutionManagementModel::
ApplicationManager
NotifyPublish
NotificationService::
StructuredPushConsumer
+ push_structured_event(StructuredEvent) : void
+ disconnect_structured_push_consumer() : void
ApplicationSupervisor
+ init(ApplicationSupervisor, SupervisorFSMDescription) : void
+ shutdown() : void

Behavior of ApplicationSupervisor and RTCs.



RTC-based system managed ApplicationSupervisor consists of RTC0 and RTC1. The system activated by an event from Application, and deactivated by an event from SensorRTC.

Comments from AB

- Two comment are returned
  - Steve Cook
  - Elisa Kendall
- Comment X-Y-Z
  - X: {Steve => 1, Elisa => 2}
  - Y: Comment number
  - Z: Comment sub-number
- Resolution
  - → Green text: Resolved issue
  - → Red text: Unresolved issue

Comment 1-1-1 Document formatting

- Headings without number
  - → Numbers are reassigned to all the headings
- Inconsistency between document and TOC
    - Platform Independent Model has no number
  - → TOC has been recreated
- Every pages has “title, version” in its footer
  - → title and version has been added: “Dynamic Deployment and Configuration for Robotic Technology Component (DDC4RTC) Specification, draft”

Comment 1-1-2
Document formatting

- The first three elements under Terms and Definitions are redundant and should be removed
  - → Removed
- The final section is called “Annex A: Title”
  - → “Title” is changed to “XML Schema and IDL”
- Not all hyperlinks in the Normative References section are active.
  - →Annex A was tagged as heading and TOC has been updated

Comment 1-1-3
Document formatting

- The Conformance clause is not a Conformance clause; it is instructions for writing a Conformance clause.
  - → Whole sentence of the section has been replaced.
  - The DEPL specification which is a basis of this specification defines enables several independent compliance points to enable different vendor implementations or user replacement of implementations. Suggested conformance points are RepositoryManager, TargetManager, NodeManager and ExecutionManager. DDC4RTC follows these conformance points of the DEPL.
  - Conformance has been revised. #2391

Comment 1-1-4
Document formatting

- Some sections start with orphan paragraphs while others start with subheadings called "Introduction". Be consistent.
  - → Subheadings "Introduction" have been removed from 8.2.1, 8.4.1 and 8.5.1
- Section 8.4 has an empty Introduction which should be removed.
  - → 8.4.1 has been removed.

Comment 1-2-1
English and grammar

- The first seven paragraphs of section 5 (7) (which is inexplicably named "Introduction" and appears as section 7 in the contents) are almost incomprehensible.
  - "Introduction" sections have been removed. #2392

- The same applies to the two paragraphs in 5.1.1 "Target Environment". The final paragraph of 5.2 has numerous grammatical errors, errors of spacing, and odd characters. (7.1.1 Target Environment)
  - Section updated. #2392

Comment 1-2-2
English and grammar

- Several of the Definitions in Terms and Definitions are too vague.
  - The definitions of SDO and RTC use different words, but I cannot discern any difference in meaning. I do not understand the definition of RT-component profile ? it is unclear what “that” is referring to. The definitions of Environment change, Deployment, Node application, Dynamic global configuration plan, and others are ungrammatical.
  - Corrected. #2393

- One you have introduced an acronym (RTC) use it consistently ? there are many references to RT component and RT-component that should all be RTC.
  - Corrected. #2394

Comment 1-2-3
English and grammar

- The section “Symbols” says “there are no special symbols or terms”. This is clearly false since the previous section defined several terms.
  - → Changed to “There are no special symbols.”
- The semantics of ApplicationSupervisor, which has no heading numbering at all, stops in the middle of a sentence.
  - The rest of sentences are copied from original document. But this section should be reviewed and revised.
  - Corrected. #2395

Comment 1-3-1 Architecture

- I understand that the submission consists of a PIM expressed in UML, which extends at least one other existing specification (DEPL). The following things are not clear:
  - The package structure of the model
  - Whether any existing specifications are changed
  - → Sentences "In this specification, the package of …" in the section 8.1 should be revised for the new DDC4RTC specification structure.
  - Updated. #2396

Comment 1-3-2 Architecture

- the section "Changes to Adopted OMG Specifications" says that DEPL is extended ? is it changed?
  - Exactly which existing specifications are reused
  - → Sentence in the 6.1 Changes to Adopted OMG Specification has been revised.
  - "This specification just extends the OMG DEPL specification without any changes in the original specification."

Comment 1-3-3 Architecture

- A later section talks about extending the “Notification Service Specification” but this is not mentioned in the Normative References
  - → Notification Service and its specification URL has been added to “3 Normative Reference”
  - “[NOT] Notification Service Specification, http://www.omg.org/spec/NOT/1.1/“
- 5.2 refers to “D & C” ? what is that? Is that an existing specification or something new?
  - → All word “D&C” and the references to D&C have been replaced with DEPL. (Five D&C -> DEPL)
- The PSM specification refers to the UML Profile for CORBA, but that is not mentioned in the Normative References.
  - UML Profile for CORBA has been added to 3 Normative References section, and all the numbers of other specifications have been updated.

Comment 1-3-4 Architecture

- In the Platform Independent Model section, the text claims that the DDC4RTC package is composed of three subpackages called Dynamic Configuration Planning, Dynamic Configuration Execution and Dynamic Configuration Metadata. This statement does not correspond with the figure, nor with the XMI.
  - → The sentence is based on old version of the specification.
  - Modified. #2396
- An earlier paragraph claimed that the DDC4RTF package consists of RTC Data Model, RTC Execution Model and Dynamic Deployment Model. This doesn’t appear to be true either. In the XMI there is a package called “other”. Overall, the package structure is poorly-documented and unclear.
  - → The sentence is based on old version of the specification.
  - 8.1 Overview should be rewritten or revised.
  - Modified. #2396

Comment 1-3-5
Architecture

- The diagrams in section 5(8) have the diagram heading “class”. According to the UML specification, this denotes a composite structure diagram. But these are not composite structure diagrams. Some of them appear to be class diagrams representing packages (in which case the header should say package P). But this is not true of ComponentAction under 5.3.3. These diagrams need to be correctly labeled and related to the model.
  - Class diagram frame has been removed.
  - Some figures with single class have been removed. #2397

Comment 1-3-6
Architecture

- There is a diagram in 5.3.4（）called “RTCDataModel”. Maybe there is supposed to be a package called RTCDataModel? But there is no such package in the XMI.
  - Figure was old version. Updated.

Comment 1-3-7
Architecture

- In 5.3.4 (8.2.6) there are 14 aggregation associations but the text only describes 11.
  - → In the 5.4.3 -> 8.2.6 RTComponentActionDescription, attributes lacks some aggregation associations: on_mode_changed, on_startup and on_shutdown

Comment 1-3-8
Architecture

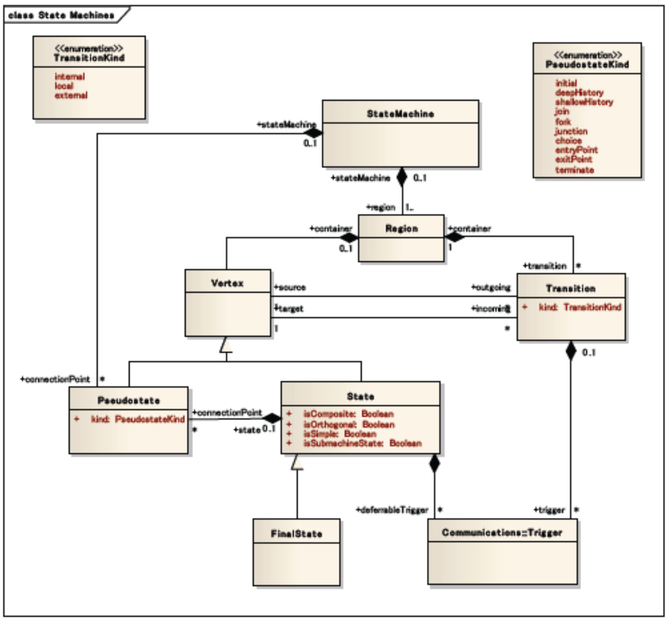
- Section 5.3.6 (changed to 8.2.8) refers to "RTC's port is same as UML component's port". What does this sentence mean? Is it actually referring to Port on UML EncapsulatedClassifier? How are the elements defined in this section supposed to relate to elements described in UML?
  - The sentence has been changed to "In the DEPL specification, a port is a kind of an interface which attached dependently to a component. On the other hand, RTC's port is based on the port owned by EncapsulatedClassifier defined in [UML2S] (for details see [RTC] section 5.2.)"
  - #2398

Comment 1-3-10
Architecture

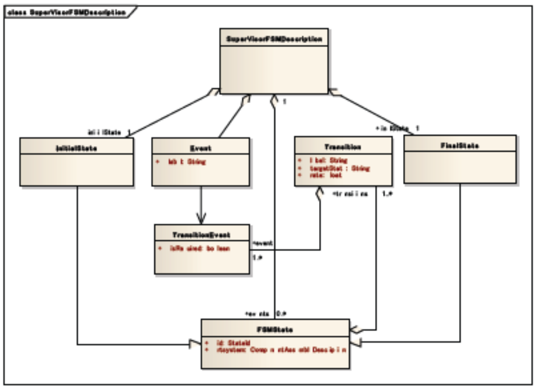
- Section 5.4 (8.4, 8.4.1 SupervisorFSM)appears to replicate part of UML's state machine metamodel. I am now confused about whether this PIM is supposed to be a metamodel or a model. If you need a state modeler to describe the SupervisorFSM, why not use UML itself? As observed above you seem to be using UML's Ports, so why not use UML's States?
  - → We would like to replace SupervisorFSM section with reference to UML' state machine meta model.

Describing DDC4RTC StateMachine by UML StateMachine metamodel

UML State Machine Metamodel



DDC4RTC State Machine Model



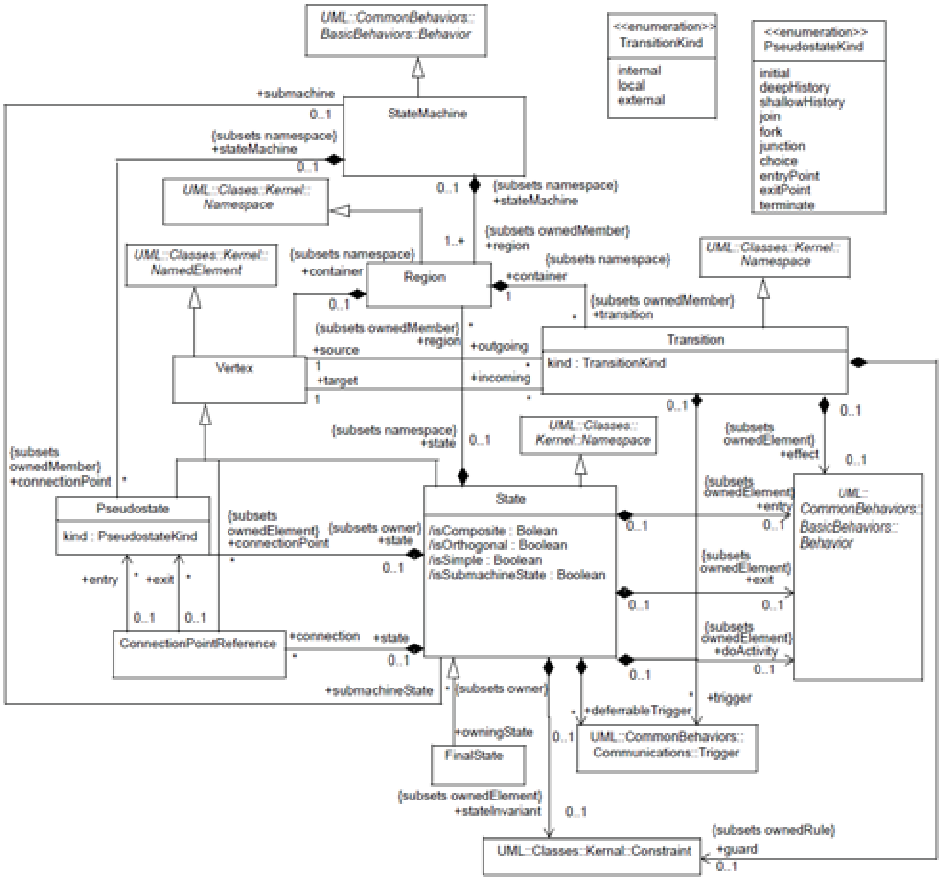

UML::CommonBehaviors::BasicBehaviors::Behavior
<<enumeration>> TransitionKind
internal
local
external
<<enumeration>> PseudostateKind
initial
deepHistory
shallowHistory
join
fork
junction
choice
entryPoint
exitPoint
terminate
+submachine
0..1
StateMachine
{subsets namespace} +stateMachine
0..1
UML::Classes::Kernel::Namespace
0..1
{subsets namespace} +stateMachine
1..*
{subsets ownedMember} +region
UML::Classes::Kernel::NamedElement
{subsets namespace} +container
Region
+container
1
{subsets namespace}
UML::Classes::Kernel::Namespace
0..1
{subsets ownedMember} +transition
{subsets ownedMember} +region
+outgoing
+source
Transition
kind : TransitionKind
Vertex
+target
+incoming
0..1
0..1
{subsets ownedElement} +effect
0..1
{subsets namespace} +state
0..1
UML::Classes::Kernel::Namespace
{subsets ownedMember} +connectionPoint
Pseudostate
kind : PseudostateKind
{subsets ownedElement} +connectionPoint
{subsets owner} +state
State
/isComposite : Boolean
/isOrthogonal : Boolean
/isSimple : Boolean
/isSubmachineState : Boolean
{subsets ownedElement} +entry
UML::CommonBehaviors::BasicBehaviors::Behavior
{subsets ownedElement} +exit
{subsets ownedElement} +doActivity
+entry
+exit
0..1
0..1
ConnectionPointReference
+connection
+state
+submachineState
{subsets owner}
+owningState
FinalState
+deferrableTrigger
+trigger
UML::CommonBehaviors::Communications::Trigger
{subsets ownedElement} +stateInvariant
0..1
UML::Classes::Kernel::Constraint
{subsets ownedRule}
+guard
0..1

Figure 15.2 - State Machines

Comment 1-3-11 Architecture

- Section 6 refers repeatedly to sections 9.x and 10.x which do not exist. It also talks about “PSM conformance points outlined in chapter 2”, which I suppose is intended to refer to the Conformance clause, which is currently numbered 1, and does not contain any conformance statements.
  - → Section 9 “Platform Specific Model” should be rewritten carefully.
  - Updated. #2400

Comment 1-3-12
Architecture

- The PSM section says “UML classes and interfaces shall be represented as IDL interfaces of the same name”. This seems to me to be a statement that belongs in another standard, not here.
  - → Section 9 “Platform Specific Model” should be rewritten carefully.
  - Updated. #2400

Comment 2-1-0
Cover page

- XMI file should be referenced; there are some blank "Source document" and "Original file(s)" references that should be eliminated.
  - → XMI file “mars/2012-02-16 (XMI)” has been added as a source document
  - Modified. #2401

Comment 2-1-1
General, throughout

- There appears to be some sort of formatting error with document footers. Also, the Table of Contents bears little resemblance to the specification itself. Is this the target for the next version of the submission?
  - → Formatting error and document footers has been corrected.

Comment 2-1-2
Preface

- 2nd paragraph needs to be rewritten.
- In the first sentence, replace "changed by physical movement of the robots, and the state transition of the application scenario" with "affected by robot movement and application or scenario state",
  - → The sentence has been replaced.
- and then replace "it is necessary to realize the dynamic deployment of components and run-time re-configuration for the robot applications“ with "it is important to be able to represent and realize dynamic component deployment and run-time re-configuration requirements".
  - → The sentence has been replaced.

Comment 2-1-3
Preface

- Also, change " further propel" to " increase", and "dynamic features in those use cases" with "requirements related to dynamic behaviors".
  - → These words have been replaced.
- 3rd paragraph, first sentence change "configuration feature for the" to "configuration requirements for" and replace "component model" with "component models".
  - → These words have been replaced.

Comment 2-1-4
Scope

- the first paragraph should talk about extending the DEPL rather than only the RTC, shouldn't it?
  - → the first paragraph is replaced with “This specification defines data models and service interfaces of deployment and configuration for RTC (OMG Robotic Technology Component Specification) based dynamic applications as an extension to DEPL (OMG Deployment and Configuration of Component-based Distributed Applications Specification) specification. “

Comment 2-1-5
Conformance

- 2nd paragraph -- change "This specification can be applied to applications which change its system structure during the application life-cycle such as robotic applications." to "This specification is designed to support development of applications whose structure changes dynamically at run-time, such as robotic applications." ... or something like that if this is too limiting. I'm not sure that life-cycle is the right term here, and it's used again in a similar way in the second sentence. Rather, I think what is meant has to do with application state and evolution over time. Perhaps this should be restated a bit more clearly, as it is an important point.
  - → The sentence has been replaced.
- Conformance needs to be specified. This is a critical omission, and needs to be addressed as soon as possible.
  - → Conformance has been specified.
- some are blank at the start of this section.
  - → Re-formatted.

Comment 2-1-6
Terms and definitions

- Also, I don't understand the definition of Domain application. Clearly this must be a contextual reference within robotics, but perhaps the definition can be expanded or rephrased a bit. I get what is meant by a multi-node application, but what does that have to do with the domain in which it is deployed? A multi-node application may or may not provide full coverage for some capability in a particular domain, in other words, but what makes it unique to that domain (where domain means a particular area of interest, or well-defined context)? This usage seems prevalent across a number of definitions, so perhaps this is jargon in robotics, but it is misleading in my view.
  - → This section should be carefully rewritten.
  - Modified. #2393

Comment 2-1-7
Introduction

- End of the first paragraph, change "such as real-time system, the system is changed dynamically are defined." to "in dynamic real-time systems is needed." 2nd paragraph needs to be rewritten, and I'm not sure exactly what to do there.
- 3rd paragraph -- change "On the other hand, in the robot system, robot and devices, in order to move in terms of network topology or physical, also dynamically change the placement of components attached to them." to "In a robotic system including robots, sensors, and/or other devices, there is a requirement for such elements to move, physically and in terms of network topology, and potentially to dynamically change internal state and component configuration."
- Change "Furthermore, it is switched by an event external scenario of robot operation, system configuration is changed dynamically due to the often it happens." to "In addition to internally triggered changes in state, location, and configuration, external environmental factors can impact operations dynamically."
- Change "That is, even during system operation as well as at the start of system operation, including redeployment and configuration of components, connections and settings between the components is carried out." to "That is, component configuration, connections, and settings established at the start of operation can be changed at run time in significant ways, either due to internal state change or external situational change.“
- Change "By applying an extension of dynamic features to support these standards DEPL, to achieve the deployment and configuration suitable for the robot using the dynamic characteristics of the RT component becomes possible." to "Support for these kinds of dynamic reconfiguration and state change requirements using RT components is possible by extending the DEPL standard as described herein."
- 4th paragraph -- delete the first sentence and add the second sentence to the end of the third paragraph.
  - → All the modifications have been applied to the Introduction section.

Comment 2-1-8
Section 5.1 (7.1)

- Dynamic Deployment and Configuration -- the first three paragraphs need a careful rewrite, which I can help with offline, as needed. The remainder of the section is quite good, however. Last paragraph, change "stat einformation" to "state information".
  - → typo has been modified, and the first section should be rewritten.
- 5.1.1 Target Environment -- needs to be re-written, and maybe should not be a separate section unless there is a 5.1.2 (none at the moment). Again, I can help with the rewrite offline, but it definitely should be made clearer.
  - → It has been rewrote. #2392

Comment 2-1-9
Section 5.2 (8.1)

- This is fairly well written. There is a parenthetical at the end of the caption under Figure 8.1, which should be 5.1? in Japanese -- please revise or delete :).
  - → Deleted.
- Also, the last sentence of the final paragraph in 5.2, "In this case refered class name is described with its name space of package." should probably be "In this case, the referenced class is includes the appropriate namespace information." or something like that.
  - → Replaced.

Comment 2-1-10
Section 5.3 Component Data Model

- not sure why the ComponentDataModel namespace is referenced within the diagram; isn't that the namespace of this model?
  - ComponentDataModel namespace is DEPL package. The diagram should be revised.
- - all figures in this section need numbers and captions
  - Numbers and Captions have been added to all the figures.
- - 5.3.2 attribute definitions need a bit of editing, and highlights should be removed
  - Done
- - ExecutionType and ActivityType are not numbered (formatting issue), and ActivityType is not referenced in any diagram
  - Done
- - 5.3.4 RTCImplementationDescription Semantics paragraph on page 24 of 43 (pdf pages, not as they are numbered) needs a bit of editing
  - Edited.
- - 5.3.6 PortInterfaceinstanceType heading moved to 5.3.7, where it is shown in the diagram? Capitalization should be corrected to PortInterfaceInstanceType, as it is in the text of the description?
  - Section "8.2.9RTCPortInterfaceDescription" and "PortInterfaceinstanceType" are replaced. Type corrected.
- - 5.3.7 Minor editing needs to be done on the Description; RTCSubcomponentPortEndPoint belongs with which diagram? Should this be in a separate section?
  - A figure has been added. It should be rewritten.

Comment 2-1-11
Section 5.4 Execution Data Model

- Introduction has no text?
  - Introduction removed.
- -- Subelements should be numbered? The descriptions are fairly thin, although I don't necessarily have an issue with this. This provides yet another finite state machine model, though -- do we have others in OMG specifications, including UML, that could be used or at least extended for this?
  - Using UML state machine meta-model should be discussed.
- -- Execution Management Model should be in a separate section? It is referred to in the text as Event Management Model, so whichever it is, there should be consistency here

- -- again, all these figures should be numbered and have captions associated with them
  - All the figures have been numbered.

Comment 2-1-12
Section 6 (9)

- -- as mentioned in my summary notes, this chapter refers to sections 9 and 10 that are not part of the specification, so it is possible that the numbering is simply "off" or that it is referring to a different specification. This should be fixed, one way or another. Also, the transformation rules are stated in text here -- is that sufficient? It seems a bit lightweight to me, or perhaps it is sufficient, but I would like to see some sort of break out of the content of the Generic Transformation Rules paragraph, and possibly a flow diagramof some sort to go with it.
  - This section should be rewritten carefully.

Comment 2-1-13
Annex A

- Annex A does not match the XMI file -- one telling example is that at the top of the diagram, where encoding is "ISO-8859-1" in the document, it is "SHIFT-JIS" in the XMI file, and a significant number of EA-specific statements are not present in the XMI in the document, so clearly they are different. They need to be reconciled. Also, it appears that there are two schema in the document, but only one XMI file -- have they been combined?
  - → XMI should be created from the newest EA.
  - **#2402**

Comment 2-1-14
XMI file

- There is quite a bit of EA-specific stuff in the XMI file, all of which needs to be removed. I inspected the XMI with a couple of different browsers, and there seems to be some EA diagram-specific material in particular that should be removed. You might try using a more recent version of EA, and exporting the XMI from that, in order to eliminate it. I have not yet had time to load this in EA and look at it in more detail, but will do so later in the week and send further comments, if any.

Robotics Domain Task Force Preliminary Agenda ver.0.0.1 robotics/2012-03-12

OMG Technical Meeting - **Cambridge, MA, USA** -- June 18-22, 2012

http://robotics.omg.org/

		TF/SIG				
		Host	**Joint (Invited**	**Agenda Item**	**Purpose**	**Room**
Monday: WG activity and Robotics-DTF Plenary(am)						
9:00	12:00			Robotics DDC4RTC submitters meeting	Arrangement	
?	?	MARS	Robotics	**Revised Submission for DDC4RTC RFP Review, Vote-to-Vote, and Voting** - Noriaki Ando (AIST) and Seung-Woog Jung (ETRI)	Joint with **MARS**	
12:00	13:00	**LUNCH**				
13:00	18:00			Architecture Board Plenary		
13:00	18:00			**DDC4RTC** (Robotic Infrastructure) WG(5h) - Noriaki Ando (AIST) and Seung-Woog Jung (ETRI)	discussion	
				RoIS (Robotic Functional Services) WG(5h): - Su-Young Chi (ETRI), Koji Kamei (JARA/ATR) and Toshio Hori (AIST)	discussion	
Tuesday: WG activity(am) and Robotics-DTF Plenary(pm)						
9:00	12:00			**DDC4RTC** (Robotic Infrastructure) WG(3h) - Noriaki Ando (AIST) and Seung-Woog Jung (ETRI)	discussion	
				RoIS (Robotic Functional Services) WG(3h): - Su-Young Chi (ETRI), Koji Kamei (JARA/ATR) and Toshio Hori (AIST)	discussion	
12:00	13:00	**LUNCH**				
13:00	13:45	Robotics		Talk: (45min) - TBA	presentation and discussion	
13:45	14:30	Robotics		Talk: (45min) - TBA	presentation and discussion	
				Break (30min)		
15:00	15:30	Robotics		Talk: (30min) - TBA	presentation and discussion	
15:30	16:10	Robotics		WG Reports and Discussion (Service WG, Infrastructure WG, Models in Robotics WG)	presentation and discussion	
16:10	16:30	Robotics		Contact Reports: - Makoto Mizukawa(Shibaura-IT), and Young-Jo Cho(ETRI)	Information Exchange	
16:30	17:00	Robotics		Robotics-DTF Plenary Wrap-up Session (**DTF Co-Chair Election**, Roadmap and Next meeting Agenda)	Robotics plenary closing	
17:00				Adjourn joint plenary meeting		
17:00	17:30			Robotics WG Co-chairs Planning Session (Preliminary Agenda for next TM, Draft report for Friday)	planning for next meeting	
Wednesday: WG activitiy						
9:00	12:00			Robotics WG activity follow-up	discussion	
12:00	14:00	**LUNCH and OMG Plenary**				
14:00	18:00			Robotics WG activity follow-up	discussion	
18:00	20:00	**OMG Reception**				
Thursday: WG activitiy						
?	?	MARS		Joint Plenary with MARS (tentative) (reserved for DDC4RTC RFP Re-Review and Voting)	Joint with **MARS**	
9:00	12:00			Robotics WG activity follow-up	discussion	
12:00	13:00	**LUNCH**				
13:00	18:00			Architecture Board Plenary		
13:00	18:00			Robotics WG activity follow-up	discussion	
Friday						
8:30	12:00			AB, DTC, PTC		
12:00	13:00	**LUNCH**				
Other Meetings of Interest						
Monday						
8:00	8:45	OMG		New Attendee Orientation		
Tuesday						
7:30	9:00	OMG		Liaison ABSC		

Please get the up-to-date version from http://staff.aist.go.jp/t.kotoku/omg/RoboticsAgenda.pdf

Robotics-DTF

Date: Friday, 22th March, 2012
Report: N. Ando
Chair:, T. Kotoku and Y. –J. Cho
URL: http://robotics.omg.org/
email: robotics@omg.org

➢Highlights from this Meeting:

Dynamic Deployment and Configuration for RTC (DDC4RTC) Submission (sponsored by MARS)

mars/2012-02-15 DDC4RTC submission
mars/2012-02-16 DDC4RTC XMI
mars/2012-02-18 DDC4RTC Inventory
mars/2012-03-26 DDC4RTC Convenient document without change bar
mars/2012-03-27 DDC4RTC Convenient document with change bar
mars/2012-03-28 DDC4RTC Errata
mars/2012-03-29 DDC4RTC XMI (revised)

Robotics Plenary: (6 participants)

- **1 Talk:** **"**A Trial Approach for Automation in Open Cut Mine" Takashi Tsubouchi (Univ. of Tsukuba) **[robotics/2012-03-05]**
- **2 WG Reports**
  - **Robotic Infrastructure WG [robotics/2012-03-06]**
  - **Robotic Functional Services WG [robotics/2012-03-07]**

Robotics-DTF

Date: Friday, 22th March, 2012
Report: N. Ando
Chair:, T. Kotoku and Y. –J. Cho
URL: http://robotics.omg.org/
email: robotics@omg.org

➢Future deliverables (In-Process):

- **Dynamic Deployment and Configuration for RTC (DDC4RTC) revised submission**

➢Next Meeting (in Cambridge):

- **Election of a Robotics-DTF Co-Chair**
- **Review of the revised submission of DDC4RTC**
- **Guest presentation**
- **Contact reports**
- **Roadmap discussion**

Minutes of the Robotics DTF Meeting - DRAFT
March 18-22, 2012
Reston, VA, USA
(robotics/2012-03-14)

Meeting Highlights

- The DDC4RTC submission was reviewed and adopted in MARS-PTF, but rejected in AB review.
- The deadline of DDC4RTC revised submission extended to the upcoming Cambridge Meeting.
- One presentation;
  - "A Trial Approach for Automation in Open Cut Mine", Takashi Tsubouchi (Univ. of Tsukuba)

List of Generated Documents

mars/2012-02-15 Dynamic Deployment and Configuration for RTC (DDC4RTC) submission
mars/2012-02-16 DDC4RTC XMI
mars/2012-02-18 DDC4RTC Inventory
mars/2012-03-25 DDC4RTC Specification Presentation (Noriaki Ando)
mars/2012-03-26 DDC4RTC Convenient document without change bar
mars/2012-03-27 DDC4RTC Convenient document with change bar
mars/2012-03-28 DDC4RTC Errata
mars/2012-03-29 DDC4RTC XMI (revised)

robotics/2012-03-01 Final Agenda (Tetsuo Kotoku)
robotics/2012-03-02 Santa Clara Meeting Minutes [approved] (Seung-woog Jung and Koji Kamei)
robotics/2012-03-03 Opening Presentation (Tetsuo Kotoku)
robotics/2012-03-04 Roadmap for Robotics Activities (Tetsuo Kotoku)
robotics/2012-03-05 A Trial Approach for Automation in Open Cut Mine (Takashi Tsubouchi)
robotics/2012-03-06 Infrastructure WG Report (Seung-Woog Jung)
robotics/2012-03-07 Robotic Functional Services WG Report (Toshio Hori)
robotics/2012-03-08 RoIS-FTF Issues and Resolutions (Toshio Hori)
robotics/2012-03-09 Call for Paper: 3rd International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR2012)
robotics/2012-03-10 Wrap-up Presentation (Tetsuo Kotoku)
robotics/2012-03-11 DDC4RTC Specification Presentation [mars/2012-03-25] (Noriaki Ando)
robotics/2012-03-12 Next Meeting Preliminary Agenda - DRAFT (Tetsuo Kotoku)
robotics/2012-03-13 DTC Report Presentation (Noriaki Ando)
robotics/2012-03-14 Reston Meeting Minutes - DRAFT (Geoffrey Biggs and Seung-woog Jung)

Minutes

Tuesday, 21 March, 2012, Lake Fairfax B, 2nd floor.
Robotics DTF Plenary Meeting

AIST, ETRI, JARA, Univ. of Tsukuba (Quorum: 3)
6 attendees

15:00 - 17:00 Robotics DTF Opening Session, Chair: Dr. Kotoku
- Minutes takers: Seung-Woog Jung (ETRI) and Geoffrey Biggs (AIST)
- Santa Clara Meeting minutes approved
    - AIST (motion), ETRI (second), JARA (white ballot)

15:00 - 15:10 Brief summary of Santa Clara meeting
- 18 participants
- 3 talks
- 2 WG reports
- Deadline of DDC4RTC revised submission was extended to the Reston meeting.
- Final report of RTC RTF was accepted.

15:10 - 15:55 Talk: A Trial Approach for Automation in Open Cut Mine
Takashi Tsubouchi (University of Tsukuba)
- Limestone mining in Japan.
- Rocks from blasting must be broken up using hydraulic breakers so they will fit into the crusher.
    - Breaking up the rocks in the bulldozing chamber is done by remote operation.
- The operator needs to move the rocks using the chisel into a good position for breaking. This is grasp-less manipulation, which is an interesting challenge for robotics.
- Studies:
    - Manipulation of rocks at the work face using a 1/12 scale electric model.
    - Detection of plugging of the grizzly bars using video cameras.

15:55 - 16:20 Infrastructure working group report (DDC4RTC)
Seung-Woog Jung, ETRI
- Submit merged document after last meeting.
- Received comments from AB members (Steve and Elisa).
- Reviewed comments on Monday and assigned tasks to make changes based on comments.
- On Tuesday, the changes were merged.
- If MARS or the AB rejects DDC4RTC, the specification will be terminated.
    - If the AB rejects with positive comments, then it will be re-submitted.

16:20 - 16:45 Service working group report (RoIS)
Toshio Hori (JARA)
- A private meeting was held in Seoul in February, 2012.
    - 26 issues were raised, 15 were resolved.
- One new issue was received by the comment deadline (2012/02/20).
- Only two members attended the OMG meeting, limiting discussions.
- Schedule from now is to present the final draft and FTF report by the next meeting deadline (21 May 2012).
- Another private meeting is planned in Seoul in May, one week before the deadline.

- Deadlines:
  - Report due date: May 21, 2012
  - Report deadline: June 29, 2012

16:45 - 17:00 Contact reports

- SIMPAR 2012, Noriaki-Ando (AIST)
  - Nov 5-8, 2012, Tsukuba, Japan

16:00 - 16:30 Robotics DTF Wrap-up Session, Chair: Dr. Kotoku

- Robotics-DTF Co-Chair : postpone voting one more meeting
- Schedule for next meeting

ATTENDEE (6attendees):

- Noriaki Ando (AIST)
- Geoffrey Biggs (AIST)
- Toshio Hori (JARA/AIST)
- Seung-Woog Jung (ETRI)
- Tetsuo Kotoku (AIST)
- Takashi Tsubouchi (Univ. of Tsukuba)

Prepared and submitted by Geoffrey Biggs (AIST) and Seung-Woog Jung (ETRI)

Date: February 2012

Dynamic Deployment and Configuration for Robotic Technology Component (DDC4RTC) Specification

Draft

OMG Document Number: mars/12-02-15

Standard document URL: http://www.omg.org/spec/DDC4RTC/1.0/PDF

Associated File(s)*: http://www.omg.org/spec/DDC4RTC/2012-xx-xx
http://www.omg.org/spec/DDC4RTC/2012-xx-xx

Source document: mars/2012-02-16 (XMI)

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

LICENSES

PATENTS

GENERAL USE RESTRICTIONS

DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c) (1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 140 Kendrick Street, Needham, MA 02494, U.S.A.

TRADEMARKS

MDA®, Model Driven Architecture®, UML®, UML Cube logo®, OMG Logo®, CORBA® and XMI® are registered trademarks of the Object Management Group, Inc., and Object Management Group™, OMG™ , Unified Modeling Language™, Model Driven Architecture Logo™, Model Driven Architecture Diagram™, CORBA logos™, XMI Logo™, CWM™, CWM Logo™, IIOP™ , MOF™ , OMG Interface Definition Language (IDL)™ , and OMG SysML™ are trademarks of the Object Management Group. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

OMG’s Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page http://www.omg.org, under Documents, Report a Bug/Issue (http://www.omg.org/technology/agreement.)

Table of Contents

Preface

Overview of this Specification

The Robotic Technology Component (RTC) Specification is an OMG standard for a component model for robotic systems. The RTC represents hardware and /or software entity that provide functionality and services for robotic systems. For deployment and configuration for component based applications, the Deployment and Configuration of Component-Based Distributed Applications (DEPL) OMG standard is available.

Generally speaking, since system structure and configuration are frequently affected by robot movement and application or scenario state, it is important to be able to represent and realize dynamic component deployment and run-time re-configuration requirements. In order to sustain and increase use of RTC and DEPL standards, it is essential to extend the DEPL standard to effectively support requirements related to dynamic behaviors.

The DDC4RTC specification defines extensions to DEPL specification to realize dynamic deployment and configuration requirements for dynamic applications based on RTC and other component models. The RTC profile defines additional component information, and the RTS Profile defines additional system information as extensions to DEPL specification. The Supervisor FSM defines a description scheme for dynamic behavior which is described as RTS Profile based state transition.

This submission provides a response to the Dynamic Deployment and Configuration for Robotic Technology Component RFP (mars/2010-06-16) and defines a solution to address the limitations identified in the afore-mentioned RFP by extending the DEPL specification with an RTC related data types, dynamic deployment and configuration, and additional services for dynamic behavior.

Intended Audience

The audience for this document is intended to be middleware developers, middleware vendors, tool vendors and application developers. This document presumes familiarity with concepts and terminology from SDO and RTC, as well as object oriented programming and component based software development. The actual data model and interface specification is provided in the Object Management Group's Interface Definition Language (IDL) and XML schema; experience with Java or C++ syntax should be sufficient to allow comprehension.

Organization of this Specification

The specification includes the following chapters:

- Scope: Specifies the scope of DDC4RTC specification covered.
- Conformance: Compliance with DDC4RTC.
- Normative References: References to other adopted specifications.
- Terms and Definitions: Formal definitions that are taken from other documents.
- Symbols: Symbols used by this specification.

- Additional Information
- DDC4RTC
- Platform independent model
- Platform specific model

OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. A Specifications Catalog is available from the OMG website at:

http://www.omg.org/technology/documents/spec_catalog.htm

Specifications within the Catalog are organized by the following categories:

OMG Modeling Specifications

- UML
- MOF
- XMI
- CWM
- Profile specifications

OMG Middleware Specifications

- CORBA/IIOP
- IDL/Language Mappings
- Specialized CORBA specifications
- CORBA Component Model (CCM)

Platform Specific Model and Interface Specifications

- CORBAservices
- CORBAfacilities
- OMG Domain specifications
- OMG Embedded Intelligence specifications
- OMG Security specifications

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
140 Kendrick Street
Building A, Suite 300
Needham, MA 02494
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: pubs@omg.org

Certain OMG specifications are also available as ISO standards. Please consult *http://www.iso.org*

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and syntax elements.

`Courier - 10 pt. Bold:` Programming language elements.

Helvetica/Arial - 10 pt: Exceptions

NOTE: Terms that appear in italics are defined in the glossary. Italic text also represents the name of a document, specification, or other publication.

1 Scope

This specification defines data models and service interfaces of deployment and configuration for RTC (OMG Robotic Technology Component Specification) based dynamic applications as an extension to DEPL (OMG Deployment and Configuration of Component-based Distributed Applications Specification) specification. It includes PIM (Platform Independent Models) and PSM (Platform Specific Models) for these models.

This specification is designed to support development of applications whose structure changes dynamically at run-time, such as robotic applications. This specification extends the existing Deployment and Configuration Specification, and utilizes the Robotic Technology Component specification as a component mode.

This specification assumes a state machine for managing when to re-deploy in the application life-cycle. Each state would be a deployment and a change in state means executing the deployment plan for that state. The developer would specify the robot's lifetime state machine, setting the deployment requirements in each state and describing the transitions.

This specification defines a management model and an information model for the dynamic deployment and configuration for Robotic Technology Components (RTC).

In particular, the specification provides:

- Ways to search for and deploy RTC into robotic systems at run-time.
- Ways to notify the relevant RTC instances of environment changes.
- Ways to search for appropriate RTC instances and dynamically configure them.

2 Conformance

The DEPL specification, which is a basis of this specification, defines several independent compliance points to enable different vendor implementations or user replacement of implementations. Suggested conformance points are RepositoryManager, TargetManager, NodeManager and ExecutionManager. DDC4RTC follows these conformance points of the DEPL.

3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

[UML2S] UML 2.4.1 Superstructure Specification, http://www.omg.org/spec/UML/2.4.1/

[UML2I] UML 2.4.1 Infrastructure Specification, http://www.omg.org/spec/UML/2.4.1/

[CORP] UML Profile for CORBA 1.0, http://www.omg.org/spec/CORP/1.0/

[MOF] MOF 2.4.1 Specification, http://www.omg.org/spec/MOF/2.4.1/

[CCM] CORBA Component Model (CCM) 3.2, http://www.omg.org/spec/CORBA/3.2/

[DDS] Data Distribution Services 1.2 Specification, http://www.omg.org/spec/DDS/1.2/

[DEPL] Deployment and Configuration of Component-based Distributed Applications Specification OMG Available Specification, http://www.omg.org/spec/DEPL/4.0/

[RTC] Robotic Technology Component specification, http://www.omg.org/spec/RTC/1.0/

[SDO] Super distributed Object Specification, http://www.omg.org/spec/SDO/1.1/

[NOT] Notification Service Specification, http://www.omg.org/spec/NOT/1.1/

4 Terms and Definitions

For the purposes of this specification, the following terms and definitions apply. The terms are defined based on the terms and definitions in DEPL, SDO and RTC specifications.

Robot application

A software application that controls a robot's behavior. Examples include a vacuum cleaning robot and a butler robot.

Super Distributed Object (SDO)

A logical representation of a hardware device or a software component that provides well-known functionality and services. One of the key characteristics in super distribution is to incorporate a massive number of objects, each of which performs its own task autonomously or cooperatively with other objects. Examples of SDOs include abstractions of devices such as mobile phones, PDAs, and home appliances, but are not limited to device abstractions. An SDO may abstract software component and act as a peer in a peer-to-peer networking system. For more details see [SDO].

Robotic Technology (RT)

Robotic Technology (RT) is a general term of the technology originating in robotics, and it can be applied not only to standalone robots but also to ubiquitous computing and other more intelligent electrical devices.

Robotic Technology Component (RTC)

A software component that supports the integration of RT systems. RTC provides provides rich component life cycle to enforce state coherency among components. It also supports fundamental design patterns including collaboration of fine-grained components tightly coupled in time, stimulus response with finite state machines and dynamic composition of components collaborating synchronously or asynchronously. For more details see [RTC].

RT System

Systems based on robotic technology, in general. Systems that are comprised of RTCs is called RT System in this specification.

Dynamic Deployment and Configuration

Changing the configuration, connections, and even utilized components of a deployed component-based system at run-time to meet the changing needs of the system.

5 Symbols

There are no special symbols.

6 Additional Information

6.1 Changes to Adopted OMG Specifications

This specification only extends the OMG DEPL specification without any changes to the original specification.

6.2 Acknowledgements

The following companies submitted this specification:

- ETRI (Electronics and Telecommunications Research Institute)
- AIST (National Institute of Advanced Industrial Science and Technology, Japan)
  AIST Tsukuba Central 2,
  Tsukuba Ibaraki 305-8568 Japan
  Contact: Noriaki Ando (n-ando@aist.go.jp)

The following companies supported this specification:

- Technologic Arts Incorporated
- Japan Robot Association

7 Introduction

The Object Management Group's Robot Technology Component (RTC) specification descibes a component model for robotic and intelligent systems, providing a framework in which such systems can be developed using model- and component-based engineering technologies. At its core is the LightweightRTC, the definition of a basic component of robot technology. It includes sophisticated introspection facilities, configuration facilities, and separates execution control from functional specification by way of Execution Contexts.

Use of any real system involves copying the components involved to the computing nodes where they will execute, configuring them according to the system's specification, and executing them. This process is known as deployment. It has been standardised at the OMG in its Deployment and Configuration of Component-based Distributed Applications specification (also known as DEPL). DEPL defines the various data models and execution models for deploying CORBA-based component systems. It is a very flexible specification. However, it is also static, being aimed at systems that do not change once deployed.

A robotic system is often a complex, distributed system, made up of the robot devices themselves as well as devices in the environment around the robots, such as sensors. Some of the devices involved are fixed while some can move, both physiically and in terms of network topology. This dynamism, which may be triggered by both internal and external events, leads to changes in internal state and the component topology necessary to achieve the robotic system's goals. The deployed component-based system must therefore change its configuration, connections, and even deployed components at runtime to meet the changing needs of the system.

The DEPL specification is a static deployment standard. However, it can be extended to meet the needs of dynamic re-deployment and re-configuration. This specification extends the OMG DEPL specification to realise functionality for dynamic system structure changes triggered by both internal and external events

7.1 Dynamic Deployment and Configuration

Dynamic deployment and configuration (DDC) refers to the monitoring of an RT System (a system comprised of RTCs) at run time, and changing its configuration (the components in use, the connections between the components, and the configuration parameters applied to the components) in response to detected events. The dynamism is discrete, in that when an event is detected, it triggers a transition from one state to another, where each state corresponds to a fixed component topology determined in advance by the developer. As part of deploying new components to a running system, the new components may need to be retrieved from a component repository on a network.

The executing system is completely specified in advance. This includes the specification of the system's various states, each of which corresponds to a component topology, and the transitions between these states, including the events that trigger them. It is the transition from one state to another that performs DDC. Events that may trigger DDC include the passing of time, changes in the lifecycle state of RTCs such as from Active to Error, data from the RTCs themselves, and hardware changes such as the removal or addition of a sensor.

An important concept in this specification is that RT Systems may, utilising the RTC specification's composite component facility, be contained within other RT Systems. This forms a tree structure of RT Systems. This same concept is used in the DEPL specification. A contained RT System is seen as a black-box component by the containing RT System, meaning that it has no knowledge of the internal structure, or even that it has an internal structure. Such isolation is necessary to make RT System reuse possible.

7.1.1 Supervisors

The core technology responsible for managing the state machine at run time, and therefore responsible for performing DDC, is the Supervisor. There is exactly one Supervisor per RT System. When one RT System is embedded in another via composition, it naturally becomes supervised (as a complete unit) by that RT System's Supervisor (this by definition means that each Supervisor, except the top-level Supervisor, has exactly one parent Supervisor). As a result, the

Supervisors also form a tree structure. This is known as a Supervision Tree, and it is a vital concept in ensuring that every Supervisor is itself supervised. It is this tree that is responsible for localising DDC, while allowing it to propagate up the tree when it cannot be handled locally.

As part of DDC, Supervisors are responsible for managing errors that occur in their RT System. The approach used in this specification is to first attempt local error handling, such as by restarting the offending component. If this fails, the error propagates up to the next Supervisor in the tree, which tries handling the error local to itself. If the error continues to fail to be handled, the Supervisor will declare a failure of its RT System, terminating all components under its control and, as its final act, notifying its parent Supervisor of the error. This propagation continues on up the tree until the top-level Supervisor is reached. If an error cannot be handled at this level, the application as a whole fails. For example, an RT-System providing a locomotion service may contain separate RT-Systems for path planning and motor control. A failure in one does not necessarily lead to a failure in the other, and can potentially be handled locally by the supervisor managing the two RT-Systems. If, on the other hand, it cannot be handled locally, the error propagates up the tree, terminating supervisors as it goes, until one is found that can handle the error.

A Supervisor is responsible for monitoring the condition of all components under its control, including composite components that may themselves contain Supervisors. The Supervisor must also monitor other pre-determined conditions, such as events from the environment. It does this via an event service interface based on the CORBA Notification Service specification.

Internally, the Supervisor functions as a Finite State Machine. Each state specifies an RT System. It begins in a known start state. This corresponds to the initial deployment plan. Events received via the notification service and RTC monitoring trigger transitions in this FSM. Each transition is a re-deployment, affected by applying the RT System of the new state. (Transitions back to the same state result in no changes and so no re-deployment.)

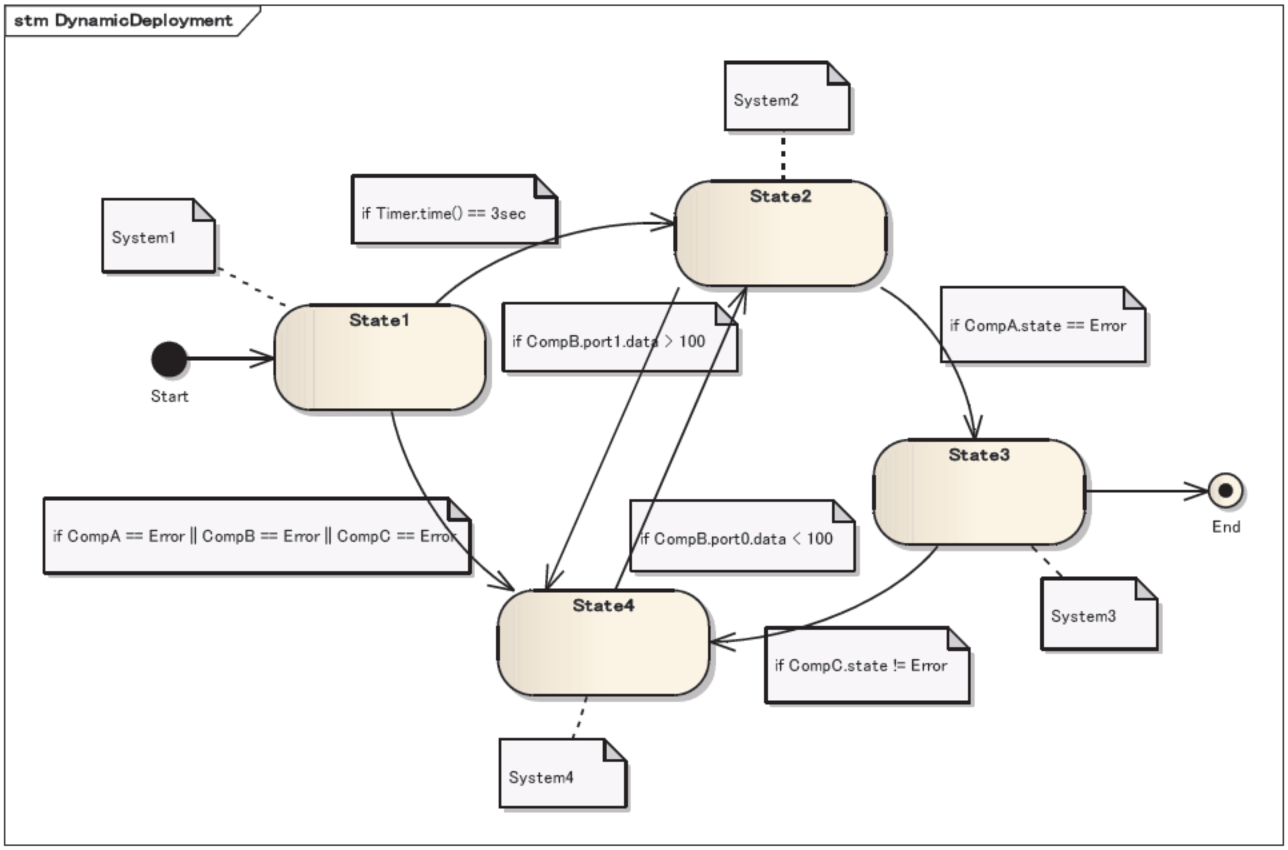


Figure 7.1:Concept of Dynamic Deployment and Configuration for RT System

Supervisors may also terminate, destroying their RT-Systems, for other reasons. If an event occurs for which the Supervisor has no suitable deployment, it will terminate with a failure. If its parent Supervisor terminates, it will terminate as well.

How the RT-System conditions are checked is not described by this specification. The Supervisor receives events over an interface; how they are generated is implementation-dependent. A possible method is to have RTCs in the RT-System that monitor the outputs of various components, and transmit events for the Supervisor to listen to. For example, one could monitor the inputs and outputs of motor controllers in an arm and send a “stalled” event when it detects no movement corresponding to the commands, leading to the Supervisor to attempt recovery. A subsequent “stalled” event may then lead to the Supervisor giving up and terminating.

In such an event-based system, the speed at which events arrive and are responded to must be given consideration. This is particularly important for repeated error events. If a Supervisor attempts error recovery, bu the error immediately reccurs, it may alternate rapidly between the error and recovery states. This is known as “bouncing.” Supervisors avoid continuous bouncing by specifying limits on state transition rates. Exceeding these limits is an error that causes the Supervisor to immediately fail.

An aspect of dynamic redeployment that is particularly important in robotics is the quantity of state information that an RTC may accumulate. Redeploying to alter a system for a new environment, for example swapping out one localisation system for another, must allow for maintaining state. Supervisors are therefore able to retrieve state from RTCs and pass it on to their successors. The developer must specify successors when creating deployments. This allows live upgrading of robot software and transitioning between related RT-Systems without losing state information. RTCs that can pass information on are placed in “lineages” by the developer. There is exactly one component in the lineage in every deployment for the RT-System. The Supervisor can shift the state from one component to another as it executes different deployments.

7.1.2 Target Environment

The target environment extends that of the DEPL specification. A robot exists in an environment that may contain additional devices it can utilise, such as sensing devices (for example, cameras) monitoring rooms of a house. In particular, if the robot is mobile, it can move through the environment, which changes what devices are available for its use.

It is not possible to anticipate the exact sequence of events an RT System will move through during operation. The developer is therefore responsible for specifying the allowed states of the RT System and the allowed transitions between those states. Once deployed in the target environment, the DDC facilities, particularly the Supervisor, are responsible for shifting the RT System between various states as appropriate to its immediate needs. This provides the flexibility and adaptability necessary.

8 Platform Independent Model

This section presents the normative specification for the platform independent deployment and configuration model.

8.1 Overview

This specification uses UML diagrams to show classes and their relationships. All classes are part of the DDC4RTC (Dynamic Deployment and Configuration for RTC) package, which contains the Component Data Model sub-package, Component Management Model sub-package, Execution Data Model sub-package and Execution Management Model sub-package. If, in a UML diagram, a class's attribute and operation compartments are suppressed, then this class is elaborated elsewhere. In this case, the diagram might also not show all of the class' associations. However, if a class is shown to have only an attribute or an operation compartment, then this signifies that the not-shown compartment is empty. I.e., if a class is shown with an attribute but no operation compartment, then the class does not have any operations.

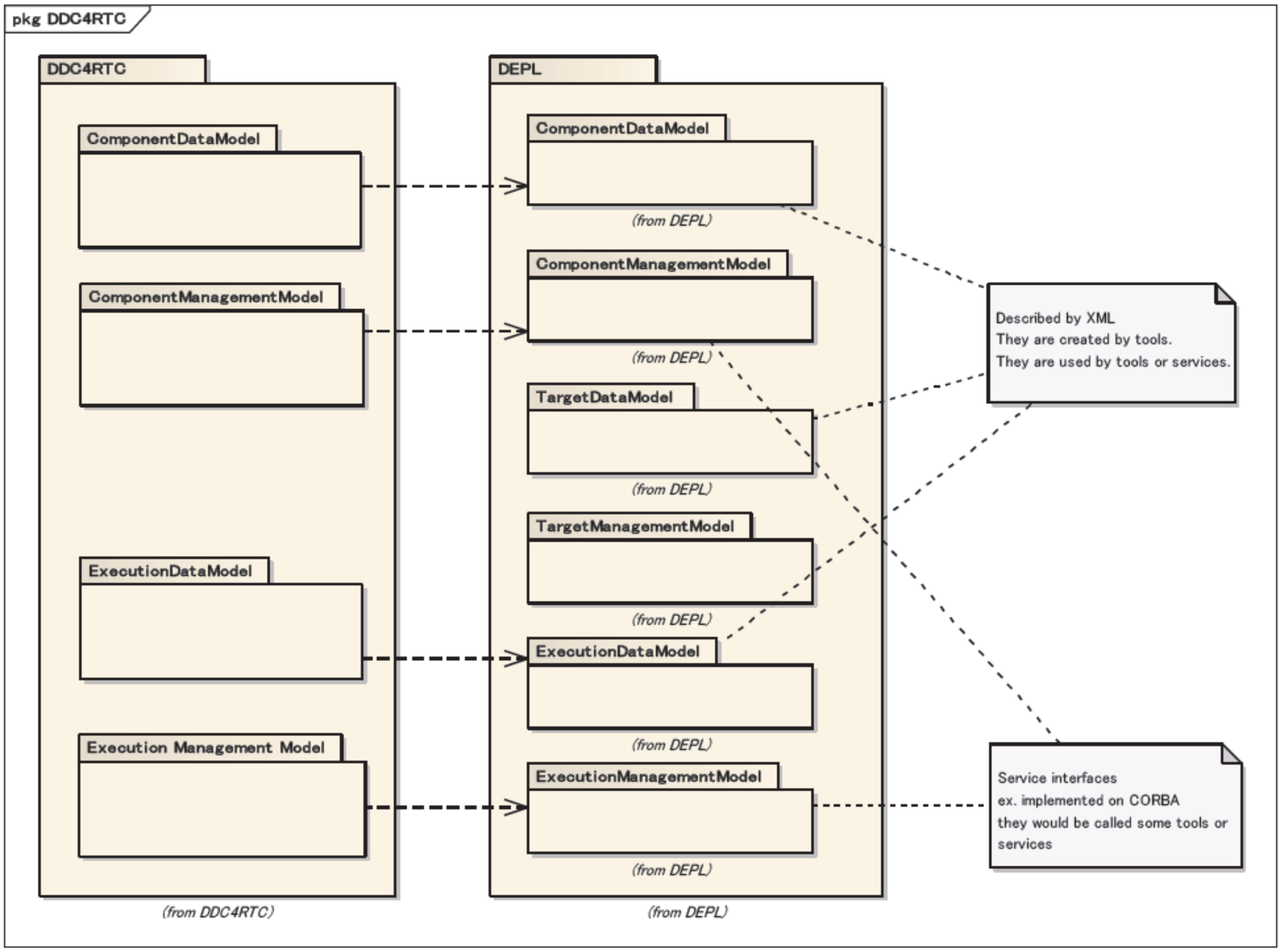


Figure 8.1:Dynamic Deployment and Configuration for RTC Model Package Structure

Each DDC4RTC sub-packages extend the same name sub-package from [DEPL] specification as shown in Figure 8.1. Some classes of DDC4RTC inherit, use, associate and aggregate DEPL classes. In this case, the referenced class is includes the appropriate namespace information.

8.2 Component Data Model

Component data model (Figure 8.2) describes profiles of component related information necessary for deployment and assembly of RT Systems. It is based on DEPL specification.

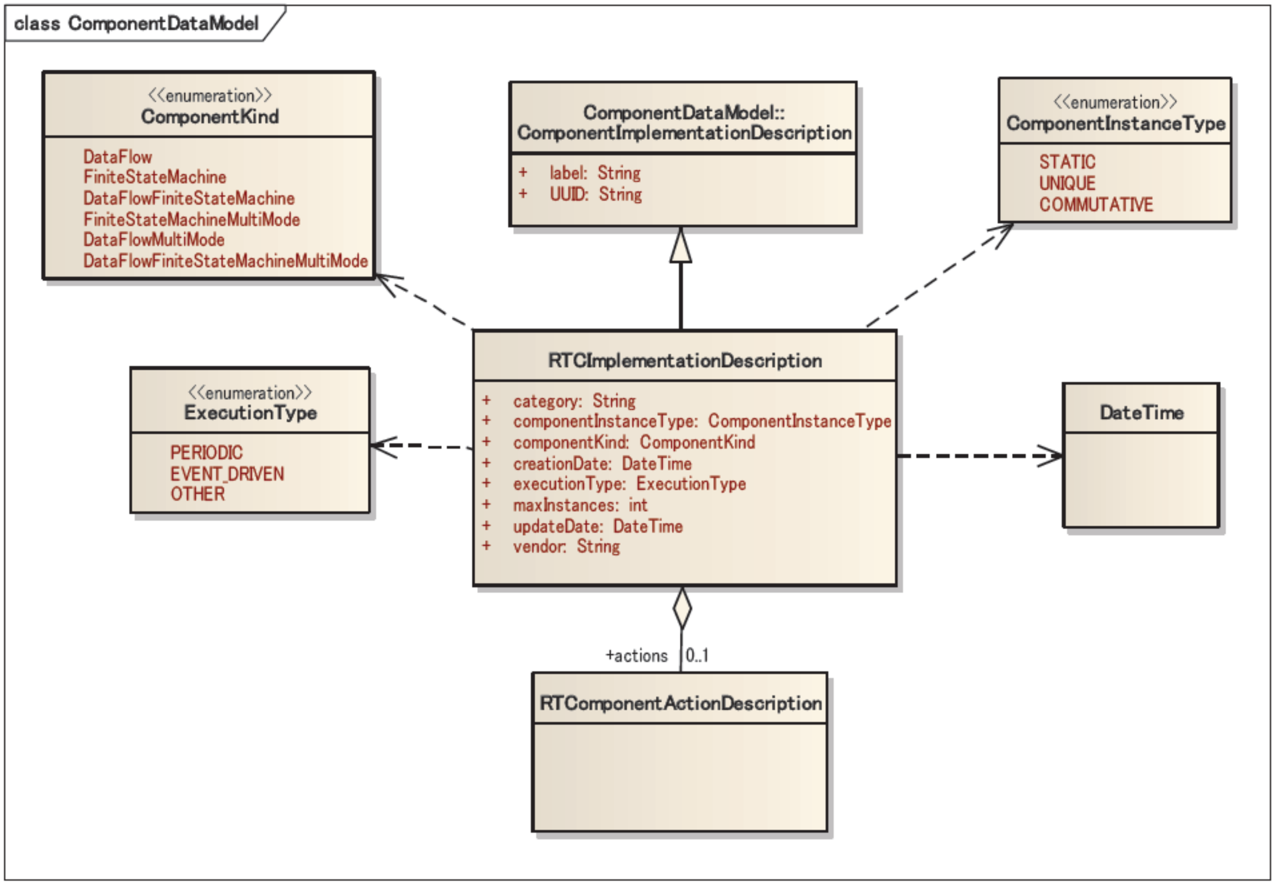


Figure 8.2:Component Data Model

8.2.1 ComponentInstanceType

Description

The ComponentInstanceType enumeration defines the component instance type of the RTCs.

Attributes

STATIC: ComponentInstanceType

With this ComponentInstanceType RTC is a component that is instantiated statically. It is a static instantiation may be performed by providing a facility for reasons such as are associated with specific hardware for, shall continue to exist throughout the life cycle of the system.

UNIQUE: ComponentInstanceType

RTC is a component with this ComponentInstnceType unique. Unique and, by maintaining a specific state data and internal, which means that it is not possible to exchange with other components of the same type at run time. Generating an instance of this type of RTC may be done dynamically.

COMMUTATIVE: ComponentInstanceType

Rtc componentInstanceType with this component is commutative. It is

	Commutative, which means that without holding the internal data and state specific, it is possible to exchange the same type of components and other always. Generating an instance of this type of RTC may be done dynamically.

8.2.2 ComponentKind

Description

The ComponentKind enumeration defines the combination of execution semantics defined in the RTC specification's execution semantics. This kind that is defined in RTC specification in “5.3 Execution Semantics” is combination of DataFlow, FiniteStateMachine and MultiMode.

Attributes

DataFlow : ComponentKind	Component of this type is the type of RTC dataFlow. This type of RTC with a type of DataFlow in ComponentAction ExecutionSemantics of [RTC].
FiniteStateMachine : ComponentKind	Component of this type is the type of RTC FiniteStateMachine. This type of RTC with atype of FiniteStateMachine ComponentAction in ExecutionSemantics of [RTC].
DataFlowFiniteStateMachine : ComponentKind	Component of this type is the type of RTC type of complex type and FiniteStateMachineDataflow. This type of RTC with FiniteStateMachine ComponentAction of both type andtype of DataFlow in ExecutionSemantics [RTC].
FiniteStateMachineMultiMode : ComponentKind	Component of this type is the type of RTC type of complex type and MultiModeFiniteStateMachine. This type of RTC with MutiMode ComponentAction of both type and type of ExecutionSemantics FiniteStateMachine in [RTC].
DataFlowMultiMode : ComponentKind	Component of this type is the type of RTC type of complex type and MultiModeDataflow. This type of RTC with MutiMode ComponentAction of both type and type ofDataFlow in ExecutionSemantics [RTC].
DataFlowFiniteStateMachineMultiMode : ComponentKind	Component of this type is the type of RTC type of complex type and MultiModeFiniteStateMachine type and Dataflow. This type of RTC with all ComponentAction inExecutionSemantics the [RTC].

8.2.3 ExecutionType

Description

A type of RTC activity.

Attributes

PERIODIC: ExecutionType	The component execution type is PERIODIC. RTC’s execution context is

PERIODIC and its execution time is bounded.

EVENTDRIVEN: ExecutionType
The component execution type is EVENT_DRIVEN. RTC’s execution context is EVENT_DRIVEN.

UNKNOWN: ExecutionType
The component execution type is UNKNOWN. RTC’s execution context is also UNKNWON.

8.2.4 ActivityType

Description

A type of RTC activity.

Attributes

PERIODIC: ExecutionType
The component execution type is PERIODIC. RTC’s execution context is PERIODIC and its execution time is bounded.

SPORADIC: ExecutionType
The component execution type is SPORADIC. RTC’s execution context is PERIODIC and its execution time is unbounded.

EVENTDRIVEN: ExecutionType
The component execution type is EVENT_DRIVEN. RTC’s execution context is EVENT_DRIVEN.

UNKNOWN: ExecutionType
The component execution type is UNKNOWN. RTC’s execution context is also UNKNWON.

8.2.5 RTComponentActionDescription

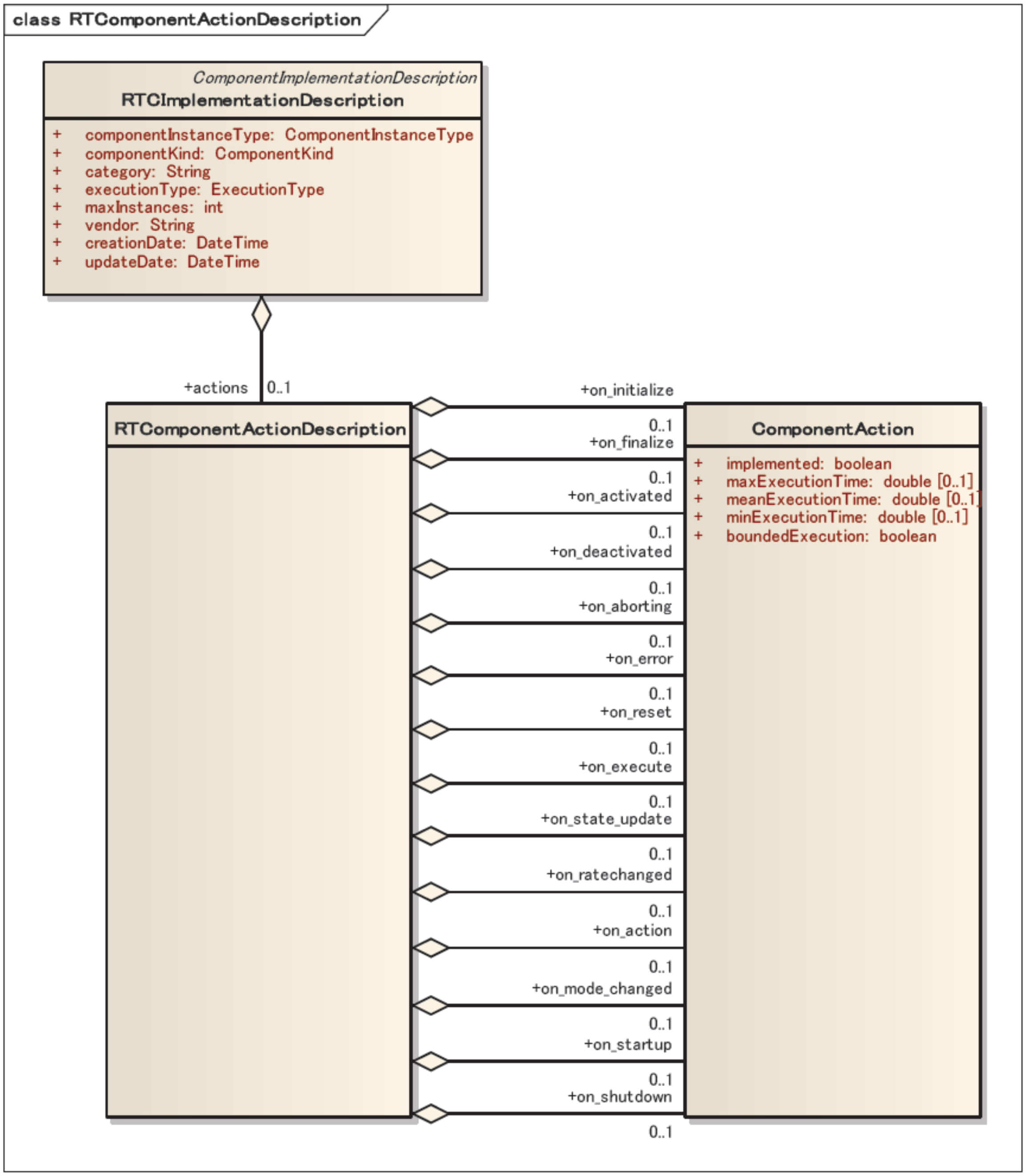


Figure 8.3:RTComponentActionDescription

Description

The RTComponentActionDescription holds the information about the callback target of ComponentAction which RTC implements.

Attributes

`on_initialize`: **ComponentAction [0..1]**	The action implementation status of on_initialize callback
`on_finalize`: **ComponentAction [0..1]**	The action implementation status of on_finalize callback.
`on_startup`: **ComponentAction [0..1]**	The action implementation status of on_startup callback.

`on_shutdown:` **ComponentAction [0..1]** The action implementation status of on_shutdown callback.

`on_activated:` **ComponentAction [0..1]** The action implementation status of on_activated callback.

`on_deactivated:` **ComponentAction [0..1]** The action implementation status of on_deactivated callback.

`on_reset:` **ComponentAction [0..1]** The action implementation status of on_reset callback.

`on_execute:` **ComponentAction [0..1]** The action implementation status of on_execute callback.

`on_state_update:` **ComponentAction [0..1]** The action implementation status of on_execute callback.

`on_rate_changed:` **ComponentAction [0..1]** The action implementation status of on_rate_changed callback.

`on_action:` **ComponentAction [0..1]** The action implementation status of on_action callback.

`on_mode_changed:` **ComponentAction [0..1]** The action implementation status of on_mode_changed callback.

`on_startup:` **ComponentAction [0..1]** The action implementation status of on_startup callback.

`on_shutdown:` **ComponentAction [0..1]** The action implementation status of on_shutdown callback.

Semantics

Whether to keep the information about the callback which will vary depending on the type of ComponentAction. Since all components of the RT with ComponentAction interface, at least with on_initialize, on_finalize, on_startup, on_shutdown, on_activated, on_deactivated, the on_reset. If the DataFlow components on_execute, on_state_update, on_rate_changed operation, FSM component on_action, MultiMode component has a o_mode_changed, RTC of a composite type with a combination of these information.

8.2.6 ComponentAction

Description

This class describes component's action call back implementation status.

Attributes

implemented: Boolean A flag whether if the action is implemented.

boundedExecution: Boolean A flag if the RTC execution time is bounded.

maxExecutionTime: double [0..1] Estimated maximum execution time of the ComponentAction of the RTC.

minExecutionTime: double [0..1] Estimated minimum execution time of the ComponentAction of the RTC.

meanExecutionTime: double [0..1]	Estimated mean execution time of the ComponentAction of the RTC.

8.2.7 RTCImplementationDescription

Description

This class inherits ComponentImplementationDescription in DEPL specification and has additional profile information of an RTC.

Attributes

componentInstanceType: ComponentInstanceType	A type of RTC instance.
componentKind: ComponentKind	A kind of the RTC. This kind that is defined in RTC specification in 5.3 Execution Semantics is combination of DataFlow, FiniteStateMachine and MultiMode.
category: String	A category of this RTC.
executionType: ExecutionKind	An execution type of this RTC. This is defined in RTC specification in section 5.2.2.7 ExecutionKind.
maxInstances: int	Maximum instance number of this RTC.
vendor: String	A vendor name of this RTC.
creationDate: DateTime	The creation date and time of this RTC.
updateDate: DateTime	The creation date and time of this RTC.

Associations

actions: ComponentActionDescription	Component’s action implementation description.

8.2.8 RTComponentPortDescription

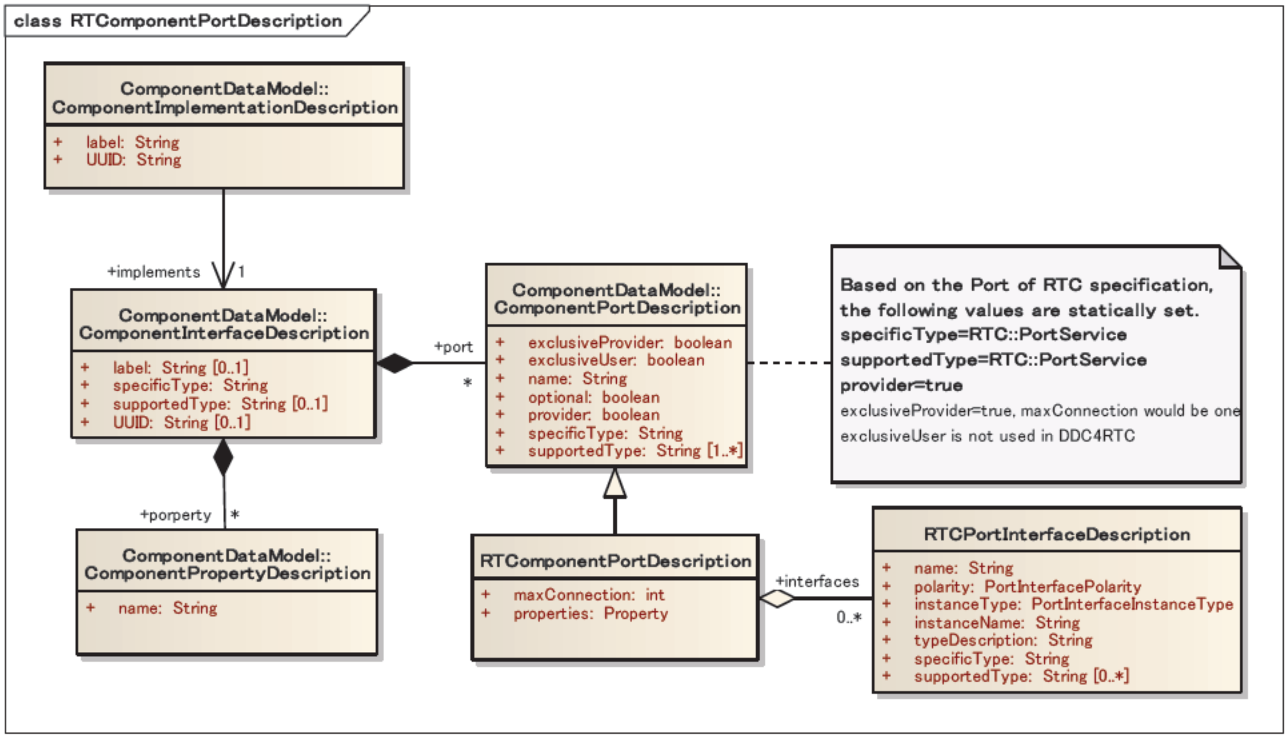

class RTComponentPortDescription
ComponentDataModel::
ComponentImplementationDescription
+ label: String
+ UUID: String
+implements
1
ComponentDataModel::
ComponentInterfaceDescription
+ label: String [0..1]
+ specificType: String
+ supportedType: String [0..1]
+ UUID: String [0..1]
+port
*
ComponentDataModel::
ComponentPortDescription
+ exclusiveProvider: boolean
+ exclusiveUser: boolean
+ name: String
+ optional: boolean
+ provider: boolean
+ specificType: String
+ supportedType: String [1..*]
Based on the Port of RTC specification,
the following values are statically set.
specificType=RTC::PortService
supportedType=RTC::PortService
provider=true
exclusiveProvider=true, maxConnection would be one
exclusiveUser is not used in DDC4RTC
+porperty
*
ComponentDataModel::
ComponentPropertyDescription
+ name: String
RTComponentPortDescription
+ maxConnection: int
+ properties: Property
+interfaces
0..*
RTCPortInterfaceDescription
+ name: String
+ polarity: PortInterfacePolarity
+ instanceType: PortInterfaceInstanceType
+ instanceName: String
+ typeDescription: String
+ specificType: String
+ supportedType: String [0..*]

Figure 8.4:RTComponentPortDescription

Description

This class describes RTC's port information including its owned port interfaces. The RTC's port can have some provided interfaces and required interfaces. The RTC's port is defined as a service of PortService interface which manages connection between ports and its owned provided/required interfaces.

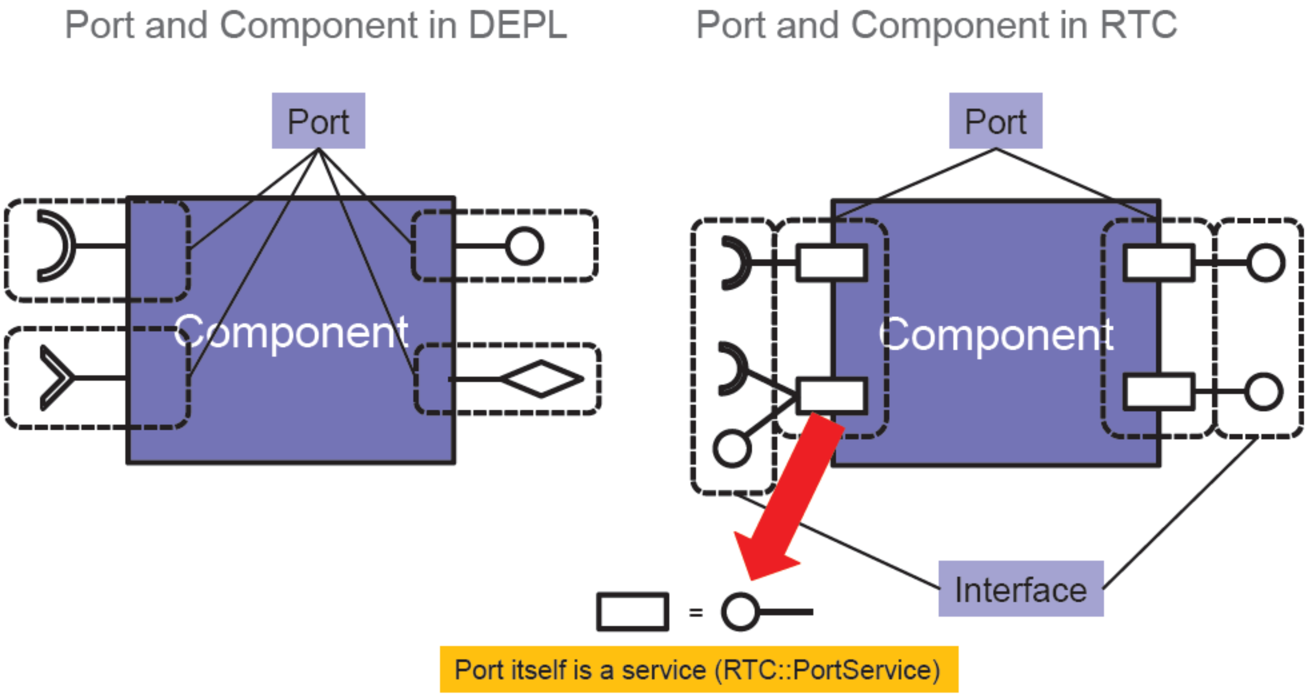

Port and Component in DEPL
Port and Component in RTC
Port
Port
Component
Component
Interface
=
Port itself is a service (RTC::PortService)

Figure 8.5:Difference between Port in DEPL component and Port in RTC

In the DEPL specification, a port is a kind of an interface which attached dependently to a component. On the other hand, RTC's port is based on the port owned by EncapsulatedClassifier defined in [UML2S] (for details see [RTC] section 5.2.) The Port is a service interface itself, and it is also place holder for provided and required interfaces. Introducing RTComponentPortDescription, this specification supports RTC's port description in the component description.

Attributes

No attributes.

Associations

serviceport: ServicePortDescription	Descriptions of service service oriented port that are owned by the target component.
dataport: DataPortDescription	Descriptions of data centric port that are owned by the target component.

Semantics

RTCPortDescription inherits DEPL::ComponentPortDescription, and some of members are fixed for RTC's PortService interface. The specifictype and supportedType are fixed to RTC::PortService, and since it always provides service, the provider member is always true.

8.2.9 RTCPortInterfaceDescription

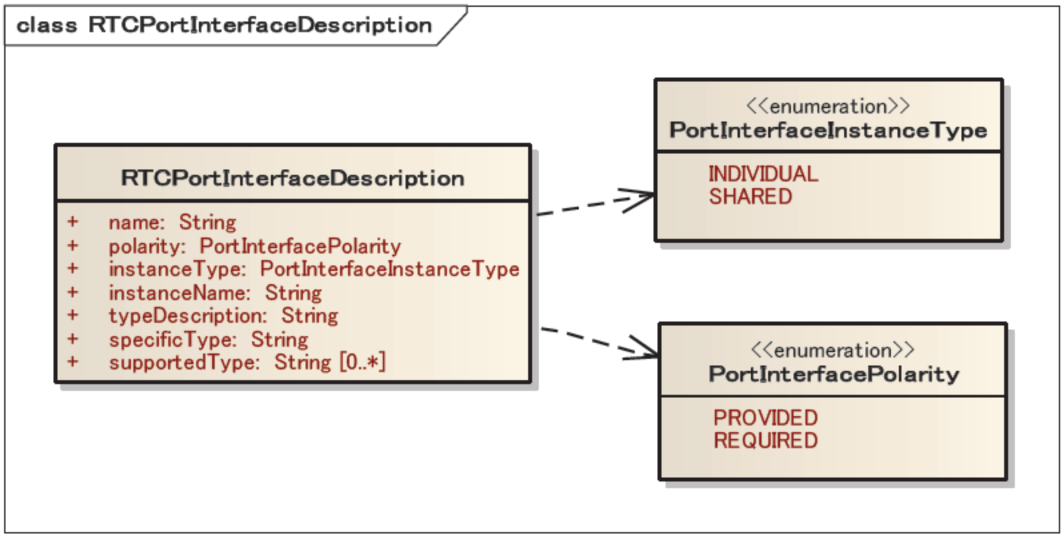

class RTCPortInterfaceDescription
RTCPortInterfaceDescription
+ name: String
+ polarity: PortInterfacePolarity
+ instanceType: PortInterfaceInstanceType
+ instanceName: String
+ typeDescription: String
+ specificType: String
+ supportedType: String [0..*]
<<enumeration>>
PortInterfaceInstanceType
INDIVIDUAL
SHARED
<<enumeration>>
PortInterfacePolarity
PROVIDED
REQUIRED

Figure 8.6:RTCPortInterfaceDescription

Description

An RTC port has zero or more interfaces. These interfaces can be a provided interfaces which provide service functions in the component, and a required interfaces that require a provided interface in the other component.

Attributes

name: **String**	Human readable interface name.
polarity: **PortInterfacePolarity**	Direction of the interface. This value can be PROVIDED or

	REQUIRED.
instanceType: **InstanceType**	The type of instance of the interface. This value can be INDIVIDUAL and SHARED. In case of INDIVIDUAL type, instance are created for each connection, and a required interface uses independent service intstance. Incase of SHARED type, only one instance is shared among required interfaces.
instanceName: **String**	The name of interface instance. instanceName should have unique name in a Port.
typeDescription:**String**	Human readable description of the interface.
specificType: **String**	The most specific type name of the interface.
SupportedType: **String [0..*]**	A list of supported interface types.

Associations

No associations.

Semantics

Interfaces belong to RTC Ports (RTCPortDescription) and its connections are managed by RTC Ports.

8.2.10 PortInterfaceInstanceType

Description

PortInterfaceInstanceType describes instance type of interface in the port. INDIVIDUAL means that instances of the port interface are created for individual connections, and SHARED means that a specific instance of the port is shared by some connections.

Attributes

INDIVIDUAL: PortInterfaceInstanceType	Instances of the port interface are created for individual connections.
SHARED: PortInterfaceInstanceType	a specific instance of the port is shared by some connections.

Associations

No associations.

8.2.11 PortInterfacePair

Description

An RTC port has zero or more interfaces. These interfaces can be a provided interfaces which provide service functions in the component, and a required interfaces that require a provided interface in the other component.

Attributes

provided: **String**	Human readable interface name.
required: **String**	Direction of the interface. This value can be PROVIDED or REQUIRED.

Associations

No associations.

Semantics

Interfaces belong to RTC Ports (RTCPortDescription) and its connections are managed by RTC Ports.

8.2.12 RTCSubcomponentPortEndPoint

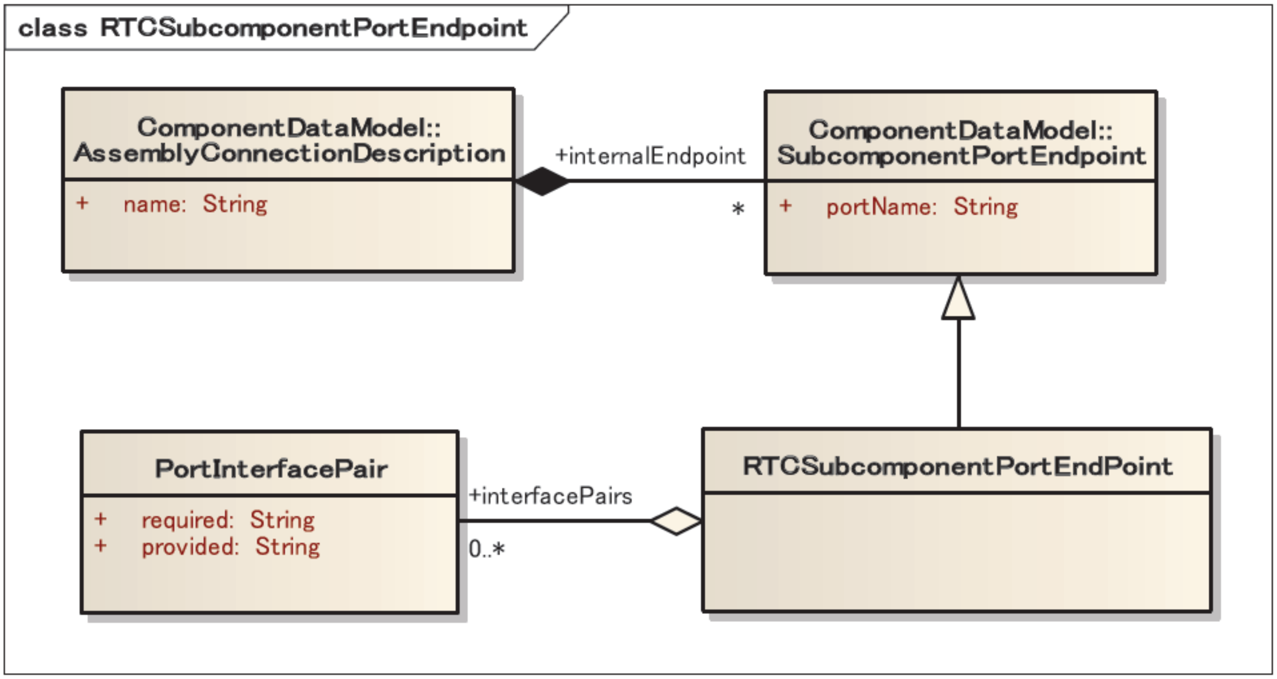


Figure 8.7:RTCSubcomponentPortEndPoint

Description

This class extend SubComponentPortEndPoint in DEPL specification.

Associations

interfacePairs: **PortInterfacePair[0..*]**	A list ofPortInterfcePairs.

Semantics

This class also replaces ComponentExternalPortEndpoint and ExternalReferenceEndpoint. in DEPL Common package.

8.3 Component Management Model

8.3.1 Repository Manager

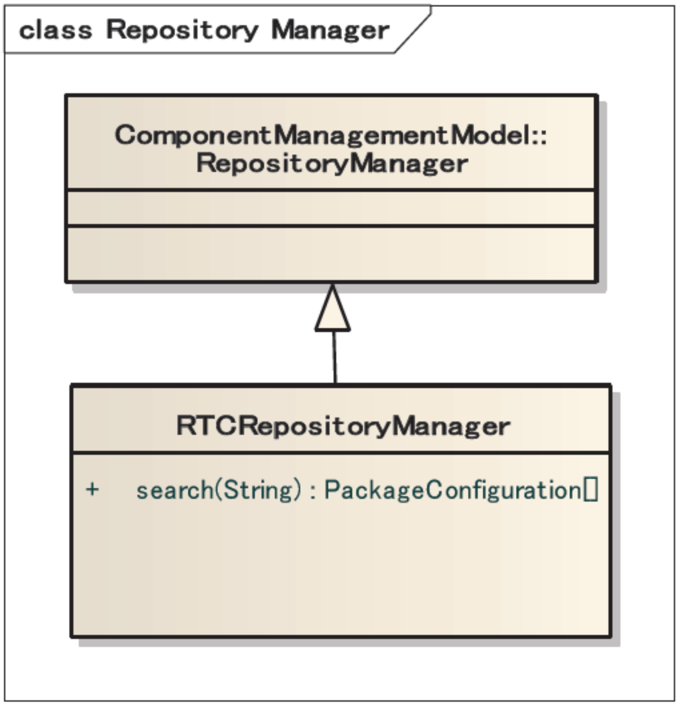

class Repository Manager
ComponentManagementModel::
RepositoryManager
RTCRepositoryManager
+ search(String) : PackageConfiguration[]

Figure 8.8:Repository Manager

Description

The RepositoryManager provides the interfaces for storing, searching, and retrieving RTCs, and the data model for the component profile description. The RepositoryManager also provides the interfaces for storing, searching, and retrieving RTC-based systems and the data model for the RTC-based system profile description.

Operations

search(query : String) : PackageConfiguration[]	This function searches a set of packages which meet the given condition and returns a sequence of PackageDescription in DEPL. The condition is given by the query which is described by the ISO/TC211 Graphic Information-filter encoding(ISO reference number : 19143).

8.4 Execution Data Model

8.4.1 SupervisorFSM

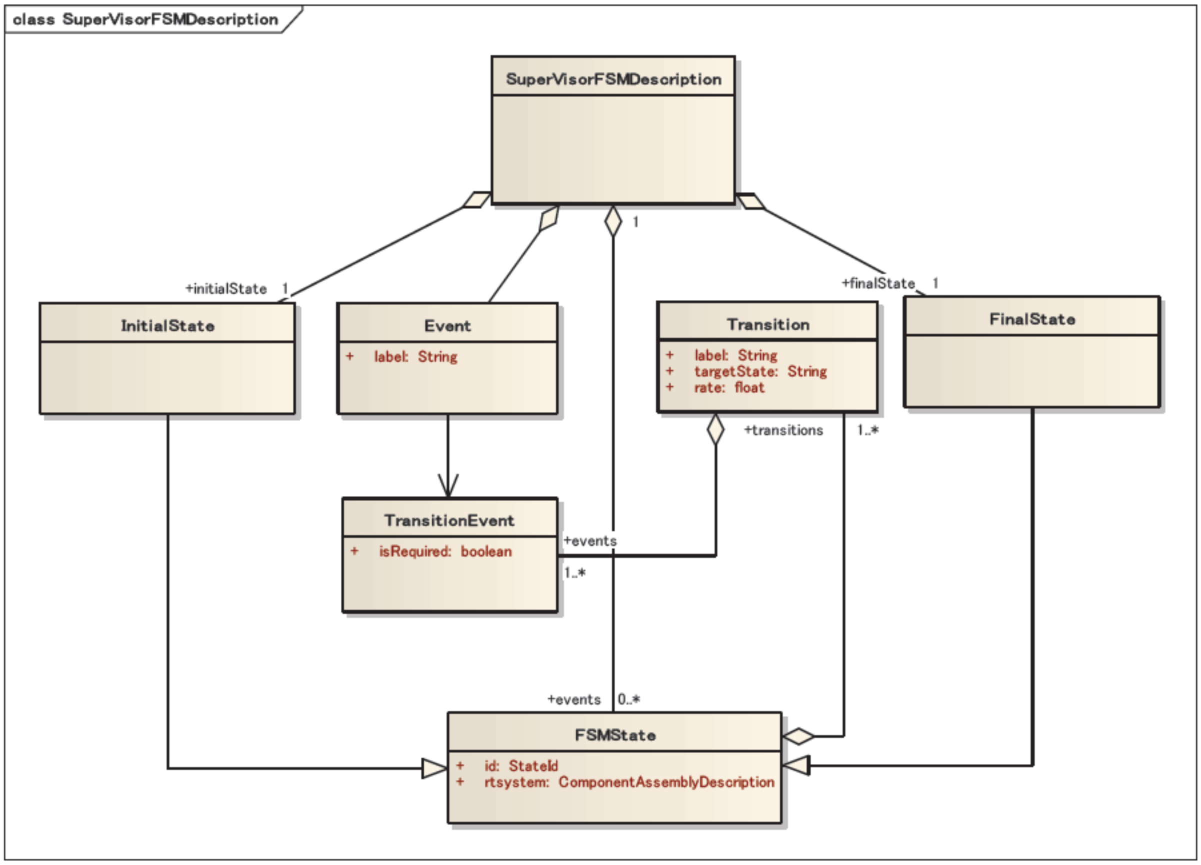


Figure 8.9:SuperVisorFSM

Description

The SupervisorFSM defines the finite state machine of an application.

Associations

initialState: InitialState [1]	A unique identifier ofr this state.
finalState: FinalState [1]	The set of non-overlapping deployments to execute on entry into this state.
state: FSMState[0..*]	The maximum rate to enter this state. For example, if the SupervisorFSM is bouncing between two states, such as an error state and its recovery state, this rate value determines how fast it can bounce before the SupervisorFSM declares a non-recoverable error and terminates.
event: Event [0..*]	

8.4.2 FSMState

Description

The State defines a state in which the SupervisorFSM may be. It corresponds to a set of one or more deployments that must be executed on entry into the state.

Attributes

label: String [1]	A unique identifier ofr this state.
rtsystems: ComponentAssemblyDescription [0..1]	The set of non-overlapping deployments to execute on entry into this state.
rate: Float	The maximum rate to enter this state. For example, if the SupervisorFSM is bouncing between two states, such as an error state and its recovery state, this rate value determines how fast it can bounce before the SupervisorFSM declares a non-recoverable error and terminates.

Associations

transitions: Transition [1..*]	A list of transitions that related this state.

Semantics

The SupervisorFSM transitions between states, executing each state's deployments as it enters. It may be limited in the rate at which it can repeatedly enter a state, in order to prevent bouncing between an error condition and recovery failure.

8.4.3 InitialState

Generalization: “State”

Description

Defines the entrance state for the SupervisorFSM

8.4.4 FinalState

Generalization: “State”

Description

Defines the terminal state for the SupervisorFSM

8.4.5 Transition

Description

A transition defines the movement from one state to another.

Attributes

label: String [1]	A unique identifier ofr this state.
targetState: String [1]	The unique identifier of the state to which this transition leads.
rate: Float	The maximum rate at which this specific transition can occur.

Semantics

A transition contains the information necessary to determine if one or more events will cause a change in state in the SupervisorFSM. If a Transition's events are determined to have occurred, then the SupervisorFSM shifts to the target state specified in the Transition.

8.4.6 Event

Description

An event that may be indicated to the SupervisorFSM.

Attributes

label: String [1]	An event that may be indicated to the SupervisorFSM..

Semantics

The SupervisorFSM is aware of all the possible events it may receive. They are stored as Events. Each Event is unique within the SupervisorFSM.

8.4.7 TransitionEvent

Description

A relation between an Event and a Transition.

Attributes

isRequired: Boolean	Determines if the referenced Event must have occured for the Transition to be valid.

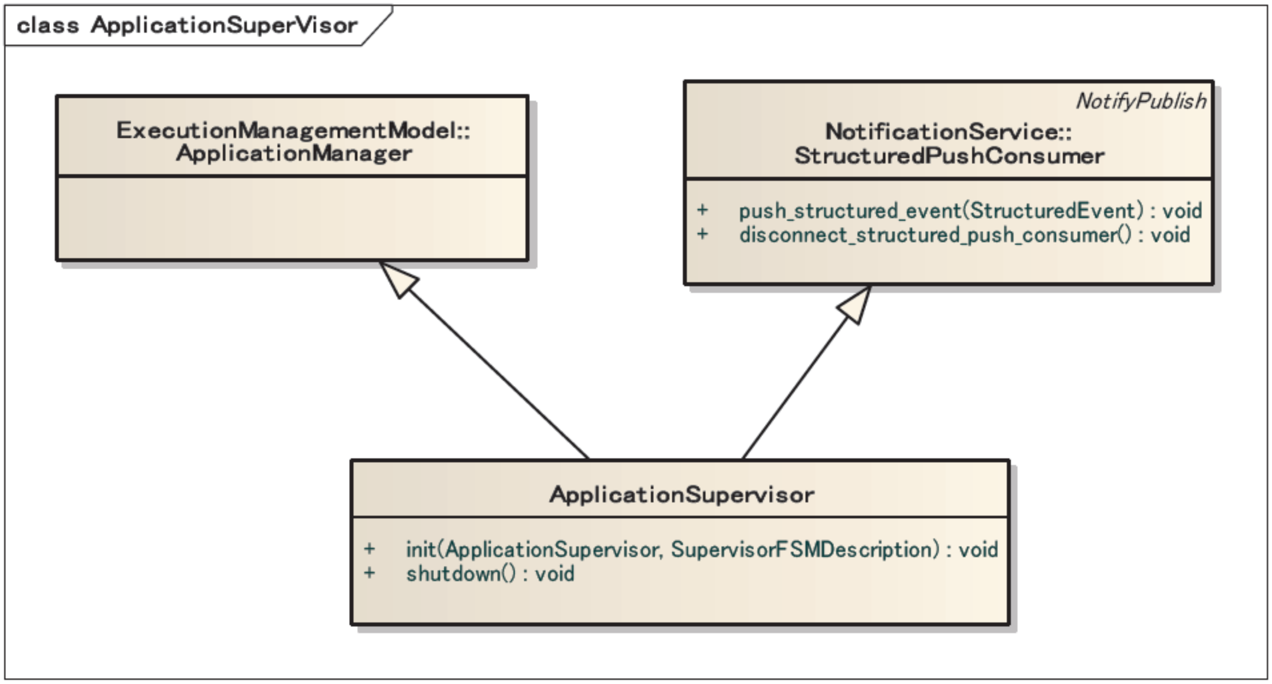
Semantics

Transitions store a list of references to the Events that may cause them to occur via the TransitionEvent. It allows Transitions to specify whether an Event must have occured for the Transition to be valid; this is roughly equivalent to specifying "and" and "or" in logic.

8.5 Execution Management Model

The Event Management Model of DDC4RTC provides certain functionality such as notifying environmental changes to RTC based applications or filtering such events based on previously registered condition. The model uses the OMG Notification Service Specification. The ApplicationSupervisor of DDC4RTC inherits the StructuredPushConsumer defined in the Notification Service Specification.

8.5.1 ApplicationSupervisor



class ApplicationSuperVisor
ExecutionManagementModel::
ApplicationManager
NotifyPublish
NotificationService::
StructuredPushConsumer
+ push_structured_event(StructuredEvent) : void
+ disconnect_structured_push_consumer() : void
ApplicationSupervisor
+ init(ApplicationSupervisor, SupervisorFSMDescription) : void
+ shutdown() : void

Figure 8.10:ApplicationSupervisor

Description

An applicationSupervisor is a unique entity within an RT System(excluding composed RT Systems, which have their own ApplicationSupervisors). It is responsible for managing the lifetimes of the RTCs that make up the RT System. The ApplicationSupervisor inherits from the DEPL ApplicationManager, for component management interfaces, and the NOT StructuredPushConsumer, for event reception.

The ApplicationSupervisor maintains several collections of objects:

- A collection of the RTCs participating in the RT System at the present time.
- A store of RTCProfiles describing the RTCs that may potentially participate in the RT System, including

those that are currently participating.

- A store of system descrption deployments and configuration that may be executed by the SupervisorFSM.

The contents of these collections is specified by the RT system packager in the RT System's specification (see the Dynamic Deployment Data Model). They are created during the first stage of deployment.

Operations

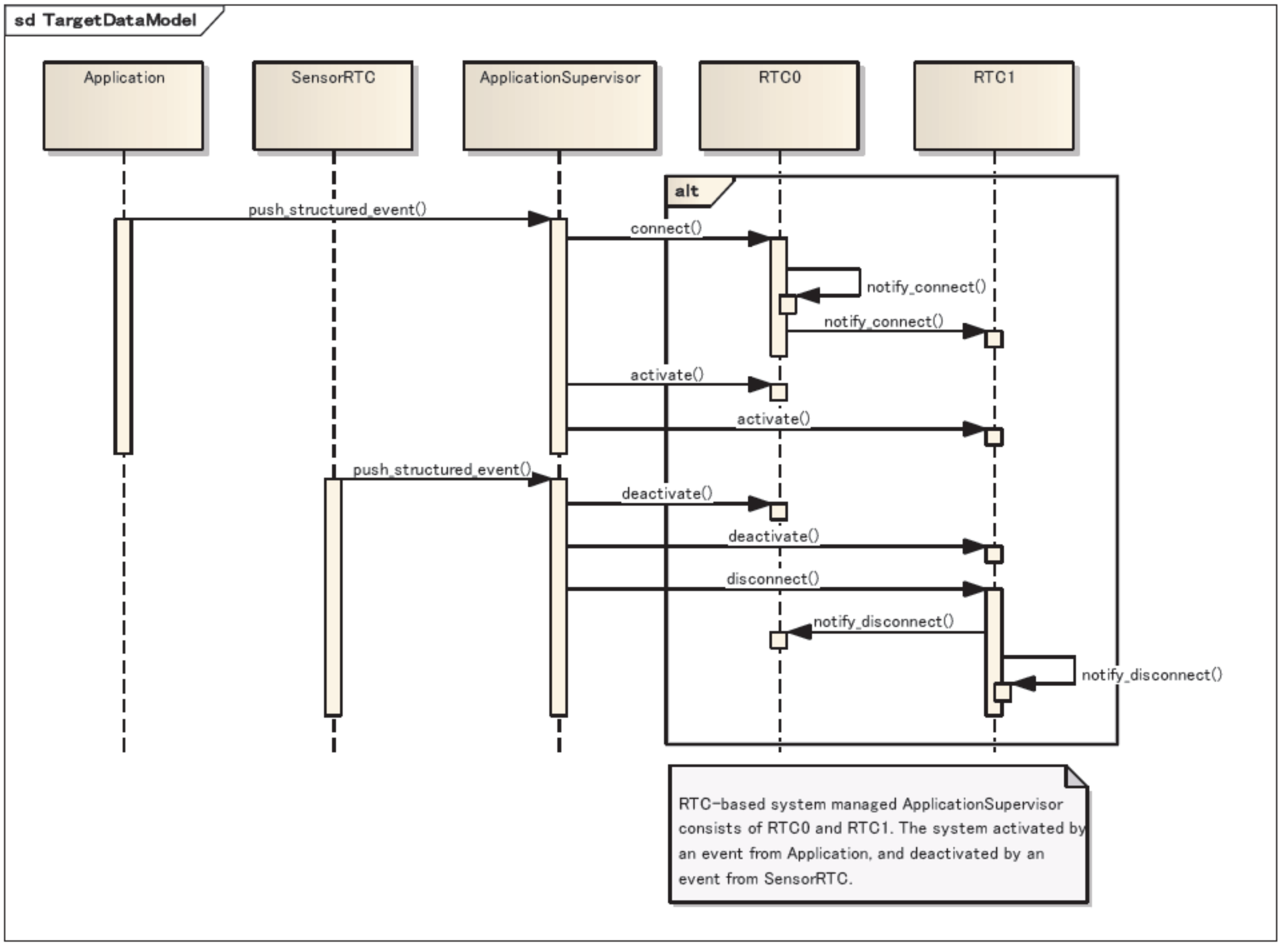
init(parent: ApplicationSupervisor, fsmdescription: SupervisorFSMDescription): void	Initialize ApplicationSupervisor with parent ApplicationSupervisor and SupervisorFSMDescription.
Shutdown(): void	Shut down the ApplicationSupervisor. This includes shutting down the RTCs it is supervising.

Generalization: “NC::StructuredPushConsumer”

Semantics

The behaviour of the ApplicationSupervisor is that of a Finite State Machine. Each state in the FSM constitutes a set of one or more non-overlapping deployments, where each deployment is specified by an ComponentAssemblyDescription, referencing RTCImplementationDescription stored by the SupervisorFSMDescription.

Upon creation, the ApplicationSupervisor will be empty. It must have a SupervisorProfile loaded into it using the loadSpecification operation. It can then be started by calling the start operation. On startup, the ApplicationSupervisor enters the initial state specified in the SupervisorProfile, executing the deployment plan(s) it specifies. Deployment follows the procedure laid out in the DEPL specification, with the following additional constraints: Components present in the new deployment and already executing in the prior deployment must not be interrupted; they must continue to run as normal. However, they should not receive execution time during the period between beginning deployment and ending deployment. Components must have the opportunity to pass state data on to replacement components.



RTC-based system managed ApplicationSupervisor consists of RTC0 and RTC1. The system activated by an event from Application, and deactivated by an event from SensorRTC.

Figure 8.11:An example behavior and communication between an ApplicationSupervisor and its RTCs.

Components are started in the exact order they are specified in the ComponentAssemblyDescription. Note that a deployment may include connections between components inside the ApplicationSupervisor's application to components in other applications. Once the initial state's deployment is complete, the ApplicationSupervisor enters a waiting state. It awaits notification of an event via the event operation. Reception of an event causes it to evaluate the current state's transitions. If a transition is valid, the ApplicationSupervisor shall transition to that state, executing the new deployment it specifies. If the transition leads to a final state, the ApplicationSupervisor shall execute its shutdown procedure. This involves removing all connections to and within the RT System and shutting down the components in the reverse order to that in which they started. As a final act before shutting down itself, the ApplicationSupervisor will notify its parent (if any) that it has shut down and the reason for shutting down. This allows the parent ApplicationSupervisor to take an appropriate action, such as restarting the ApplicationSupervisor in case of an error, replacing the shut down RT system with an alternative RT System, or propagating the shutdown reason further up the Supervision Tree (particularly in the case of an error). Notification to the parent is performed via the parent's event operation.

The ApplicationSupervisor is required to handle all events. If an event not specified in the SupervisorFiniteStateMachineDescription is received, the ApplicationSupervisor terminate with an error. For the set of pre-defined events shown below, it should respond as described. For all other events, it should respond as described by its SupervisorFiniteStateMachineDescription.

- CHILD_SHUTDOWN_ERROR: A child of the ApplicationSupervisor has shut down after an error. The ApplicationSupervisor must respond by restarting the child component according to its SupervisorFiniteStateMachineDescription. This may optionally include shutting down all children started after the failed component (according to the start up order), or all children in the RT System. Which option is used is defined in the ComponentAssemblyDescription. Any other children shut down must be restarted with the failed child in the appropriate order as defined in the ComponentAssemblyDescription. When other children are shut down, they must be informed that it is due to a failed component.
- CHILD_SHUTDOWN_OK: A child of the ApplicationSupervisor has shut down after completing execution. Response, if any, is determined by the SupervisorFiniteStateMachineDescription.

The ApplicationSupervisor must track the number of times an error occurs. If the error occurs at a rate greater than the maximum defined in the SupervisorFiniteStateMachineDescription, the ApplicationSupervisor must terminate with an error, including terminating all of its children. For example, an error repeatedly occurring at 5Hz in an ApplicationSupervisor with a configured maximum of 1Hz will trigger termination.

8.5.2 Relation to the DEPL ApplicationManager

The ApplicationSupervisor inherits from the ApplicationManager. It reuses the interface for starting and destroying applications, but with more specific semantics (described above).

8.5.3 DirectoryManager

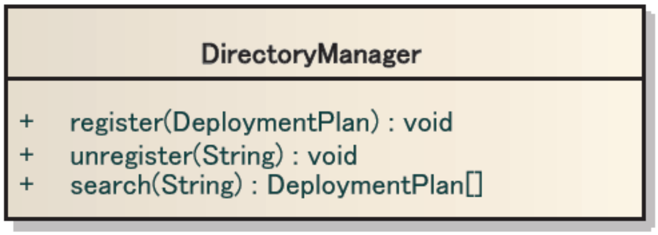

DirectoryManager
+ register(DeploymentPlan) : void
+ unregister(String) : void
+ search(String) : DeploymentPlan[]

Figure 8.12:DirectoryManager

Description

The DirectoryManager provides the interfaces for RTC instance discovery and the data model which describes the RTC instance.

Operations

register(desc: RTCInstanceDeploymentDescription): void	This function registers the information of an RTC instance to the directory under the given information of the RTC instance. It throws ALREADY_REGISTERED when the RTC instance is already registered, INVALID_ ARGUMENT when the given information of the RTC instance is not correct, and UNKNOWN_ERROR when there is some error occurred.
unregister(ref : String) : void	This function deletes the information of an RTC instance from the directory. It throws NOT_REGISTERED when the RTC instance is not registered, INVALID_ ARGUMENT when the ref is not correct, and UNKNOWN_ERROR when there is some error

	occurred.
search(query : String) : sequence<RTCInstanceDeploymentDescription>	This function searches a set of RTC instances which meet the given condition and returns a sequence of RTCInstanceDeploymentDescription. The condition is given by the query which is described by the ISO/TC211 Graphic Information-filter encoding(ISO reference number : 19143). It throws INVALID_ ARGUMENT when the query is not correct, and UNKNOWN_ERROR when there is some error occurred.

Associations

managedInfos: RTCInstanceDeploymentDescription[0..*]	The DirectoryManager manages a set of RTCInstanceDeploymentDescription.

Semantics

No semantics.

9 Platform Specific Models

In order to maximize interoperability, this document describes one PSM that should be considered normative in the section 9.2. The PSM draws on a common set of IDL definitions.

9.1 UML-to-IDL Transformation

The PSM require IDL definitions for the interfaces, data types, and other model elements from the PIM. They also require IDL representations of the model elements from [UML] on which the PIM depends: RepositoryManager, TargetManager, NodeManager, ExecutionManager, ApplicationSupervisor, Directorymanager. Representing all of the UML in IDL is beyond the scope of this specification. This specification takes a more parsimonious approach.

- IDL definitions for the elements from this specification are provided explicitly in Section 9.2.
- Mapping rules from a subset of UML to IDL are provided in this section. Only those parts of UML that are necessary to describe the PIM are described here. Mappings of all other UML constructs are implementation-defined.

9.1.1 Basic Types and Literals

The standard UML String and Boolean types are also used by this specification.

- Boolean: boolean
- String: string
- int: long

The literal specifications defined in the PIM —as well as those referenced from [UML]—shall be represented as IDL literal values.

9.1.2 Classes and Interfaces

UML classes and interfaces shall be represented as IDL interfaces of the same name.

- Each operation or attribute on the UML classifier shall be represented by a corresponding operation or attribute in IDL.
- The general classifiers of UML classes and interfaces shall be represented as inheritance between the corresponding IDL interfaces.

9.1.3 Enumerations

Enumerations in the PIM shall be represented as IDL enumerations of the same name. Each attribute shall correspond to a constant within that enumeration.

9.1.4 Packages

A UML package described in the section 8.1 shall be represented by an IDL module of the same name.

9.2 CORBA PSM

In this PSM, DDC4RTC is mapped to CORBA interfaces extending the relevant IDL interfaces described in Section 9.1.

9.2.1 Generic Transformation Rules

The mapping to IDL is accomplished using the rules set forth in the UML Profile for CORBA. To enable the usage of an index, the composition of the target element in its container is qualified with the "ordered" constraint. Wherever the multiplicity of an attribute, parameter or return value is not exactly one (but 0..1, 1..* or *), a new class is introduced to represent a sequence of the type of the attribute, parameter or return value. The sequence class has the «CORBASequence» stereotype, and its name is the english plural of the name of the type. The sequence class has a 128 composition association with the element class that is navigable from the sequence to the element. The composition is qualified with the index of the sequence. The attribute, parameter or return value is then replaced with an attribute, parameter or return value, respectively, with the same name as before, but with the type being the newly introduced sequence class and the exactly one (1..1) multiplicity. A similar rule is applied to all navigable association or composition ends whose multiplicity is not exactly one (but 0..1, 1..* or *): a new class is introduced to represent a sequence of the class at the navigable end; this sequence class is defined as describe above. The original association or composition end is then replaced with a navigable association or composition end, with the same role name as before, at the new sequence class, with a multiplicity of exactly one (1..1). According to the rules in the UML Profile for CORBA, these associations and compositions will then map to a structure member in IDL, its type being a named sequence of the referenced type. Excepted from the two rules above are attributes, parameters, return values or navigable association or composition ends where the type is String, unsigned long or Endpoint. Instead of defining new sequence types, the existing types in the CORBA package are being used; see below. Note that in combination, these rules map non-composite associations between classes with a common owner and a multiplicity other than 1 to sequence of "unsigned long" type. Another exception from the rule above are attributes of type String with the 0..1 (zero or one) multiplicity. In this case, the multiplicity is updated to 1..1 (exactly one). If the value is missing in an XML representation of the model, the empty string is used as default value.

9.2.2 Sequence of String

A type representing a sequence of strings already exists in the CORBA package and can be re-used. Wherever the String type is used with a multiplicity other than exactly one, it is mapped to the StringSeq class from the CORBA package as shown above. It then maps to the CORBA::StringSeq type in IDL (from the orb.idl file).

9.2.3 Primitive Types

The UML data types String, Integer and Boolean are mapped to the classes string, long and boolean in the CORBAProfile package, respectively. They will then map to the string, long and boolean types in IDL, respectively.

9.2.4 Mapping to IDL

After applying the transformations defined in this section, IDL is generated by applying the rules set forth in the UML Profile for CORBA specification [UPC].

9.2.5 DEPL

The ComponentDataModel, ComponentManagementModel, ExecutionDataModel, ExecutionManagemenModel relies on the DEPL specification. Implementations that support that packages shall use that specification's CORBA PSM.

9.2.6 Notification Service

The StructuredPushConsumer relies on the Notification Service specification [NOT]. Implementations that support that packages shall use that specification's CORBA PSM.

Annex A: XML Schema and IDL

(normative)

A.0 CORBA IDL

```
#include <DEPL.idl>
#include <COS/NotificationService.idl>

#pragma prefix "omg.org"

#define PackageConfiguration string
#define RTCInstanceDeploymentDescription string

module DDC4RTC
{
  typedef sequence<PackageConfiguration> PackageConfigurationList;

  interface RepositoryManager
    : DEPL::RepositoryManager
  {
    PackageConfigurationList search(in string query);
  }

  interface ApplicationSupervisor
    : DEPL::ApplicationManager, NotificationService::StructuredPushConsumer
  {
    void init(in ApplicationSupervisor parent,
              in SupervisorFSMDescription fsmdescription);
    void shutdown();
  };

  typedef sequence<RTCInstanceDeploymentDescription> RTCInstanceDeploymentDescriptionList;

  interface Table1 DirectoryManage {
    void register(in RTCInstanceDeploymentDescription desc);
    void unregister(in string ref);
    RTCInstanceDeploymentDescriptionList search(in string query);
  };
};
```

A.1 Component Data Model XML Schema

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:Deployment="http://www.omg.org/Deployment">
	<xs:import namespace="http://www.omg.org/Deployment"
schemaLocation="D:\00\OMG\DEPL\Spec_Defined_Deployment.xsd"/>
	<xs:include/>
	<xs:simpleType name="PortInterfacePolarity">
		<xs:restriction base="xs:string">
```

```
            <xs:enumeration value="PROVIDED"/>
            <xs:enumeration value="REQUIRED"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="PortInterfacePair" type="PortInterfacePair"/>
    <xs:complexType name="PortInterfacePair">
        <xs:sequence>
            <xs:element name="required" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="provided" type="xs:string" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="RTCImplementationDescription" type="RTCImplementationDescription"/>
    <xs:complexType name="RTCImplementationDescription">
        <xs:complexContent>
            <xs:extension base="ComponentImplementationDescription">
                <xs:sequence>
                    <xs:element name="componentInstanceType"
type="ComponentInstanceType" minOccurs="1" maxOccurs="1"/>
                    <xs:element name="componentKind" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="category" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="executionType" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="maxInstances" type="xs:int" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="vendor" type="xs:string" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="creationDate" type="xs:dateTime"
minOccurs="1" maxOccurs="1"/>
                    <xs:element name="updateDate" type="xs:dateTime" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="actions" type="RTComponentActionDescription"
minOccurs="0" maxOccurs="1"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:simpleType name="ComponentInstanceType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="STATIC"/>
            <xs:enumeration value="UNIQUE"/>
            <xs:enumeration value="COMMUTATIVE"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="ActivityType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="PERIODIC"/>
            <xs:enumeration value="EVENT_DRIVEN"/>
            <xs:enumeration value="SPORADIC"/>
            <xs:enumeration value="MIXED"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:simpleType name="ComponentKind">
        <xs:restriction base="xs:string">
            <xs:enumeration value="DataFlow"/>
            <xs:enumeration value="FiniteStateMachine"/>
            <xs:enumeration value="DataFlowFiniteStateMachine"/>
```

```
                        <xs:enumeration value="FiniteStateMachineMultiMode"/>
                        <xs:enumeration value="DataFlowMultiMode"/>
                        <xs:enumeration value="DataFlowFiniteStateMachineMultiMode"/>
                </xs:restriction>
        </xs:simpleType>
        <xs:simpleType name="ExecutionType">
                <xs:restriction base="xs:string">
                        <xs:enumeration value="PERIODIC"/>
                        <xs:enumeration value="EVENT_DRIVEN"/>
                        <xs:enumeration value="OTHER"/>
                </xs:restriction>
        </xs:simpleType>
        <xs:element name="DateTime" type="DateTime"/>
        <xs:complexType name="DateTime">
                <xs:sequence/>
        </xs:complexType>
        <xs:element name="ComponentAction" type="ComponentAction"/>
        <xs:complexType name="ComponentAction">
                <xs:sequence>
                        <xs:element name="implemented" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                        <xs:element name="maxExecutionTime" type="xs:double" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="meanExecutionTime" type="xs:double" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="minExecutionTime" type="xs:double" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="boundedExecution" type="xs:boolean" minOccurs="1"
maxOccurs="1"/>
                </xs:sequence>
        </xs:complexType>
        <xs:element name="RTComponentActionDescription" type="RTComponentActionDescription"/>
        <xs:complexType name="RTComponentActionDescription">
                <xs:sequence>
                        <xs:element name="on_activated" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_deactivated" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_aborting" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_error" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_reset" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_execute" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_state_update" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_ratechanged" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_action" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_initialize" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_finalize" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
                        <xs:element name="on_mode_changed" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
```

```
            <xs:element name="on_startup" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
            <xs:element name="on_shutdown" type="ComponentAction" minOccurs="0"
maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="RTComponentPortDescription" type="RTComponentPortDescription"/>
    <xs:complexType name="RTComponentPortDescription">
        <xs:complexContent>
            <xs:extension base="ComponentPortDescription">
                <xs:sequence>
                    <xs:element name="maxConnection" type="xs:int" minOccurs="1"
maxOccurs="1"/>
                    <xs:element name="properties" type="Deployment:Property"
minOccurs="1" maxOccurs="1"/>
                    <xs:element name="interfaces" type="RTCPortInterfaceDescription"
minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="RTCPortInterfaceDescription" type="RTCPortInterfaceDescription"/>
    <xs:complexType name="RTCPortInterfaceDescription">
        <xs:sequence>
            <xs:element name="name" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="polarity" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="instanceType" type="PortInterfaceInstanceType" minOccurs="1"
maxOccurs="1"/>
            <xs:element name="instanceName" type="xs:string" minOccurs="1"
maxOccurs="1"/>
            <xs:element name="typeDescription" type="xs:string" minOccurs="1"
maxOccurs="1"/>
            <xs:element name="specificType" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="supportedType" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:simpleType name="PortInterfaceInstanceType">
        <xs:restriction base="xs:string">
            <xs:enumeration value="INDIVIDUAL"/>
            <xs:enumeration value="SHARED"/>
        </xs:restriction>
    </xs:simpleType>
    <xs:element name="RTCSubcomponentPortEndPoint" type="RTCSubcomponentPortEndPoint"/>
    <xs:complexType name="RTCSubcomponentPortEndPoint">
        <xs:complexContent>
            <xs:extension base="SubcomponentPortEndpoint">
                <xs:sequence>
                    <xs:element name="interfacePairs" type="PortInterfacePair"
minOccurs="0" maxOccurs="unbounded"/>
                </xs:sequence>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
</xs:schema>
```

A.2 Execution Data Model Schema

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:Deployment="http://www.omg.org/Deployment">
    <xs:import namespace="http://www.omg.org/Deployment"
schemaLocation="D:\00\OMG\DEPL\Spec_Defined_Deployment.xsd"/>
    <xs:element name="FSMState" type="FSMState"/>
    <xs:complexType name="FSMState">
        <xs:sequence>
            <xs:element name="id" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="rtsystem" type="Deployment:ComponentAssemblyDescription"
minOccurs="1" maxOccurs="1"/>
            <xs:element name="transitions" type="Transition" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="SuperVisorFSMDescription" type="SuperVisorFSMDescription"/>
    <xs:complexType name="SuperVisorFSMDescription">
        <xs:sequence>
            <xs:element name="events" type="FSMState" minOccurs="0"
maxOccurs="unbounded"/>
            <xs:element name="finalState" type="FinalState" minOccurs="1" maxOccurs="1"/>
            <xs:element name="initialState" type="InitialState" minOccurs="1" maxOccurs="1"/>
            <xs:element name="Event" type="Event" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="Event" type="Event"/>
    <xs:complexType name="Event">
        <xs:sequence>
            <xs:element name="label" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="TransitionEvent" type="TransitionEvent" minOccurs="1"
maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="Transition" type="Transition"/>
    <xs:complexType name="Transition">
        <xs:sequence>
            <xs:element name="label" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="targetState" type="xs:string" minOccurs="1" maxOccurs="1"/>
            <xs:element name="rate" type="xs:float" minOccurs="1" maxOccurs="1"/>
            <xs:element name="events" type="TransitionEvent" minOccurs="1"
maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
    <xs:element name="InitialState" type="InitialState"/>
    <xs:complexType name="InitialState">
        <xs:complexContent>
            <xs:extension base="FSMState">
                <xs:sequence/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="FinalState" type="FinalState"/>
    <xs:complexType name="FinalState">
        <xs:complexContent>
            <xs:extension base="FSMState">
```

```
                <xs:sequence/>
            </xs:extension>
        </xs:complexContent>
    </xs:complexType>
    <xs:element name="TransitionEvent" type="TransitionEvent"/>
    <xs:complexType name="TransitionEvent">
        <xs:sequence>
            <xs:element name="isRequired" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
```