ICRES 2022: 7th International Conference on Robot
Ethics and Standards, Seoul, South Korea, 18-19 July 2022.
https://doi.org/10.13180/icres.2022.18-19.07.019

An Ethical Black Box for Social Robots: a draft Open Standard

Alan F.T. Winfield[1⋆], Anouk van Maris[1], Pericle Salvini[2], and Marina Jirotka[2]

[1] Bristol Robotics Lab, University of the West of England, Bristol
[2] Department of Computer Science, University of Oxford

Abstract. This paper introduces a draft open standard for the robot equivalent of an aircraft flight data recorder, which we call an ethical black box. This is a device, or software module, capable of securely recording operational data (sensor, actuator and control decisions) for a social robot, in order to support the investigation of accidents or near-miss incidents. The open standard, presented as an annex to this paper, is offered as a first draft for discussion within the robot ethics community. Our intention is to publish further drafts following feedback, in the hope that the standard will become a useful reference for social robot designers, operators and robot accident/incident investigators.

Keywords: ethical black box, social robots, traceability, transparency, robot ethics, responsible robotics

1 Introduction

In [8] we argued the case that robots and autonomous systems should be equipped with the equivalent of an aircraft Flight Data Recorder to continuously record sensor and relevant internal status data. We call this an ethical black box (EBB). We argued that an ethical black box will play a key role in the processes of discovering why and how a robot caused an accident, and thus an essential part of establishing accountability and responsibility.

We propose that the EBB needs a standard specification. A standard specification has several benefits. First, a standard approach to EBB implementation in social robots will greatly benefit accident and incident (near miss) investigations [9]. Second, an EBB will provide social robot designers and operators with data on robot use that can support both debugging and functional improvements to the robot. Third, an EBB can be used to support robot 'explainability' functions to allow, for instance, the robot to answer 'Why did you just do that?' questions from its user. And fourth, a standard allows EBB implementations to be readily shared and adapted for different robots and, we hope, encourage manufacturers to develop and market general purpose robot EBBs.

This paper is structured as follows. Section 2 provides a brief recap of the history of data loggers, and associated standards, in aviation, critical infrastructure

⋆ alan.winfield@brl.ac.uk

and road vehicles. In section 3 we give a brief high-level description of the EBB, and in section 4 we argue the case for a standardised EBB, and a draft open standard as a starting point. In section 5 we conclude the paper by outlining the draft open standard in Annex A.

2 A brief introduction to Data Loggers

The term 'black box' was first used informally in the late 1940s for navigational instruments, within the Royal Air Force. The term was then extended to cover any kind of apparatus within a sealed container. From the mid 1960s the colloquial use of the 'black box' has narrowed to refer to the Flight Data Recorder (FDR), now fitted as standard in aircraft.

Black box – or flight data recorders – were introduced in 1958, for larger aircraft, and since then have vastly expanded in scope in what flight data they record. Initially FDRs included time navigation data about the position of surfaces and the pilots' movement of controls; latterly sensor data on the internal and external environment as well as the functioning of components and systems are also recorded, alongside autopilot settings such as selected headings, speeds, altitudes and so on [2]. FDRs on modern aircraft record more than 1000 parameters.

The transfer of the black box concept into settings other than aviation is not new. Data loggers for critical infrastructure such as Supervisory Command and Data Acquisition (SCADA) systems are also standard practice [5, 6]. The largest deployment of black box technology outside aviation is within the automobile and road haulage industries for data logging [7, 10]. Data loggers for vehicles are generally known as Event Data Recorders (EDRs). Standards for EDRs include IEEE 1616 'Standard for Motor Vehicle Event Data Recorder (MVEDR)' first published in 2004 and revised in 2021 [1].

3 The Ethical Black Box

All robots collect sense data, and – on the basis of that sense data and some internal decision making process (AI) – send commands to actuators. This is of course a simplification of what in practice will be a complex set of connected systems and processes but, at an abstract level, all intelligent robots will have the three major subsystems shown in blue, in Fig. 1. A social robot is no different, except that it is designed to interact directly with humans.

We define the ethical black box (EBB) as a system for securely recording date- and time-stamped operational data from a social robot. We use the term *ethical black box* to emphasise our view that to deploy social robots without an EBB would be irresponsible.

The Ethical Black Box (EBB) and its data flows, shown in red in Fig. 1 will need to collect and store data from all three robot subsystems: sensor data, actuator demands and actual positions, and robot decisions — ideally with the reasons for those decisions. All of these data will need to be date and time

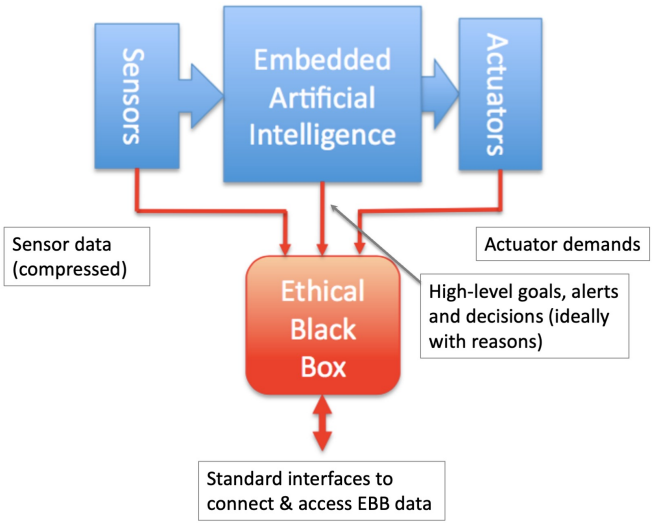

Figure 1. Robot sub-systems with an Ethical Black Box and key dataflows.

stamped. An important property of the EBB is that the data flows from the robot to the EBB are strictly one-way. It is important that the EBB is a passive sub-system, accepting data from the robot controller, and not affecting the robot's operation.

4 Why do we need a standardised EBB?

We contend that social robots should not only be fitted with an EBB, but that EBB should follow a standard specification. There are several benefits from a standard EBB:

1. A standard approach to EBB implementation in social robots will greatly benefit accident and incident (near-miss) investigations. In [9] we argue that social robots bring greater risks than industrial robots, and hence the likelihood of harms — including psychological, societal or environmental harms — is greater. Without the data provided by the EBB the investigation of accidents or near-miss incidents in order to discover what happened, why it happened and how to prevent it happening again is difficult, if not impossible.
2. An EBB will provide social robot designers and operators with data on robot use that can support both debugging and functional improvements to the robot. Thus it becomes a powerful diagnostic tool during research and development.

3. An EBB can be used to support robot explainability functions to allow, for instance, the robot to answer "Why did you just do that?" questions from its user [4].[3] And
4. a standard allows EBB implementations to be readily shared and adapted for different robots and, we hope, both encourage manufacturers to develop and market general purpose robot EBBs, and regulators to require EBBs.

Bruce Perens, creator of The Open Source Definition, outlines a number of criteria an open standard must satisfy, including:

1. "Availability: Open standards are available for all to read and implement.
2. Maximize End-User Choice: Open Standards create a fair, competitive market for implementations of the standard.
3. No Royalty: Open standards are free for all to implement, with no royalty or fee.
4. No Discrimination: Open standards and the organizations that administer them do not favor one implementor over another for any reason other than the technical standards compliance of a vendor's implementation.
5. Extension or Subset: Implementations of open standards may be extended, or offered in subset form."[4]

5 The Draft Open Standard

In Annex A we set out the first draft of an Open Standard for an EBB for social robots. Following the model of the Internet open standards this draft is a Request for Comments (RFC). Subsequent drafts will incorporate feedback. The standard is written following BS 0:2021 *A Standard for Standards* [3].

Annex A sets out the requirements for an EBB Software Module that could be integrated within a robot's controller, or as a stand alone software process connected, via a network connection, to the robot.

Annex A details the normative requirements for the data structures and formatting. These data structures fall into three categories: Meta Data, which stores information about the robot that the EBB is connected to, Data Data, which stores information on the number of records and dates and times of the oldest and most recent records in the EBB, and Robot Data, which stores operational data on the robot. The final category Robot Data will comprise most of the data stored in the EBB. All three structures are date- and time-stamped and have a common format. Annex A also provides examples of records for each of these three data structures together with an example of a complete set of EBB records, and how often particular robot data records might be written.

The overall aim of the standard is to provide a technical specification for an EBB for Social Robots. This specification, alongside an open-source library of model implementations, will provide a resource to enable developers to build an EBB into their robots.

[3] Noting that this would require the robot's control system to access it's own EBB - and thus violate the principle of the EBB's passivity.

[4] https://opensource.com/resources/what-are-open-standards

Acknowledgments

This work has been conducted within project RoboTIPS: Developing Responsible Robots for the Digital Economy supported by EPSRC grant ref EP/S005099/1.

References

1. IEEE standard for motor vehicle event data recorder (mvedr). *IEEE Std 1616-2021 (Revision of IEEE Std 1616-2004)*, pages 1–184, 2021.
2. D. R. Grossi. Aviation recorder overview, national transportation safety board [NTSB]. *J. Accid. Investig.*, 2(1):31–42, 2006.
3. British Standards Institute. BS 0 2021, a Standard for Standards — principles of standardization. *BS 0 2021*, pages 1–42, 2021.
4. Vincent J. Koeman, Louise A. Dennis, Matt Webster, Michael Fisher, and Koen Hindriks. The "why did you do that?" button: Answering why-questions for end users of robotic systems. In Louise A. Dennis, Rafael H. Bordini, and Yves Lespérance, editors, *Engineering Multi-Agent Systems*, pages 152–172, Cham, 2020. Springer International Publishing.
5. Thomas Morris and Kalyan Pavurapu. A retrofit network transaction data logger and intrusion detection system for transmission and distribution substations. In *2010 IEEE International Conference on Power and Energy*, pages 958–963, 2010.
6. Jeong Seok Oh. A practical study on data logger for gas industry. In James J. Park, Vincenzo Loia, Gangman Yi, and Yunsick Sung, editors, *Advances in Computer Science and Ubiquitous Computing*, pages 860–864, Singapore, 2018. Springer Singapore.
7. P. R. Thom and C. A. MacCarley. A spy under the hood: controlling risk and automotive EDR. *Risk Manag. Mag*, 55(2):22—26, 2008.
8. A. F. Winfield and M. Jirotka. The case for an ethical black box. In Y. Gao, S. Fallah, Y. Jin, and C. Lekakou, editors, *Towards Autonomous Robotic Systems (TAROS 2017) Lecture Notes in Computer Science Vol. 10454*, pages 262–273. Springer, Cham, 2017.
9. Alan F. T. Winfield, Katie Winkle, Helena Webb, Ulrik Lyngs, Marina Jirotka, and Carl Macrae. Robot Accident Investigation: A case study in responsible robotics. In Ana Cavalcanti, Brijesh Dongol, Rob Hierons, Jon Timmis, and Jim Woodcock, editors, *Software Engineering for Robotics*. Springer, Cham, 2021.
10. M. Worrell. Analysis of bruntingthorpe crash test data, impact. *J. Inst. Traffic Accid. Investigators*, 21(1):4–10, 2016.

Annex A: Draft Standard for an Ethical Black Box Software Module for Social Robots

Request For Comments, Draft 0.1

A.1 Scope

This draft open standard sets out normative technical requirements for a data logger for Social Robots, which we call an Ethical Black Box (EBB). This draft is

a Request For Comments (RFC), inviting feedback and suggestions for improvements. Further drafts will incorporate revisions in response to this feedback.

The aim of the standard is to provide researchers, developers and operators with a technical specification for an EBB for Social Robots. This specification, alongside an open source library of model implementations, will provide a resource to enable developers to build an EBB into their robots.

The standard sets out the specification for a software module for an EBB, which may be implemented either as a software module alongside the robot's control system, or as a stand alone software process within a system connected (i.e. via a network connection) to the robot. This module may also be implemented within a hardware EBB physically connected to the robot, although the specification of the hardware is outside the scope of this draft.

A.2 Normative References

- BS 0:2021, *A standard for standards — Principles of standardization*
- BS ISO 8373:2021, *Robotics — Vocabulary*

A.3 Terms and Definitions

For the purposes of this document, the following terms and definitions apply.

Ethical Black Box (EBB): A system for securely recording date- and time-stamped operational data from a social robot.

Social Robot: An intelligent service robot designed to interact with humans. For definitions of intelligent service robot refer to BS ISO 8373:2021.

A.4 EBB Normative Requirements for Data

This section sets out requirements for the data that should be captured by the EBB.

EBB data organisation The data stored in the EBB shall comprise three types of record: meta data, data data and robot data. The EBB shall contain one meta data (MD) record only, one data data (DD) record only, and a number of robot data (RD) records. The maximum number of RD records shall be fixed for each EBB according to the limits of its storage capacity. The general organisation of EBB data is shown in Table 1.

The EBB RD logs shall be written in order starting from RD record 1, and proceeding to RD record n. After record n has been written the EBB shall write the next RD record overwriting RD record 1, and write subsequent RDs in record 2, and so on. In this way the EBB always stores the most recent set of RD records, up to its maximum capacity of n records.

Meta Data, Data data and Robot Data records have the same overall structure, as shown in 2.

Each EBB record consists of a 2 character record labels 'MD', 'DD' or 'RD' followed by a variable number of fields.

EBB records
Meta data (MD) record
Data data (DD) record
Robot data (RD) record 1
Robot data (RD) record 2
...
Robot data (RD) record n

Table 1. General organisation of EBB data

Record	length	format
Record type 'MD', 'DD' or 'RD'	2 chars	ASCII text
Number of fields and chars in record	12 chars	ASCII 000:00000000
Date record written	10 chars	ASCII yyyy:mm:dd
Time record written	12 chars	ASCII hh:mm:ss:ms
Data record 1	variable	see below
...		
Data record m	variable	see below
Checksum	4 chars	ASCII

Table 2. Common structure of Meta Data, Data Data and Robot Data Records

Each field in a record consists of a 4 character label, followed by data elements defined according to the label. EBB fields are defined below for each of the 3 types of record. Note that several fields are common to all 3 record types.

The EBB Meta Data Record The Meta Data Record shall store information about the robot that the EBB is fitted to (name, version or model no, and serial no), the robot's developer/manufacturer and operator, the contact details of the person responsible for the robot, and information on the EBB itself.

Table 3 defines each field in the MD record, and includes both required and optional fields. Here a string is defined as a variable length ASCII sequence terminated by the null character ASCII \0.

Notes on the MD fields:

1. The record size *recS*, has a 12 char data element formatted as 3 numeric chars and 8 numeric chars separated by a colon, i.e. 000:00000000. It follows that the maximum permissible number of fields in a record is 999, and the maximum number of characters is 99,999,999. Note also that the size and character count must include the recS field.
2. The EBB date field *ebbD* is 12 ASCII characters with colon separated year, month, and day, i.e. yyyy:mm:dd.
3. The EBB time field *ebbT* is 10 ASCII characters with colon separated hour, minute, second and millisecond, i.e. hh:mm:ss:ms.
4. The record checksum *chkS* shall be computed using a 64-bit non-cryptographic hash function, to be determined.

An example of a complete Meta Data Record is shown in 4.

label	data	length	requirement
recS	record size, field and chars, including recS field	12 chars	required
ebbD	EBB date record written	10 chars	required
ebbT	EBB time record written	12 chars	required
botN	robot name	string	required
botV	robot version no	string	optional
botS	robot serial no	string	optional
botM	robot manufacturer	string	required
opeR	robot operator	string	optional
resP	name and contact details of responsible person	string	required
ebbN	EBB name and version no	string	required
chkS	checksum for complete record	8 hexadecimal chars	required

Table 3. Meta Data Fields

field	comment
MD	record label
recS 010:00000000	number of fields:chars in record
ebbD 2022:04:20	date 20 April 2022
ebbT 16:40:20:000	time 16:40 and 20.000 seconds
botN NAO\0	NAO robot
botV 4\0	v4
botM Aldebaran\0	Manufacturer
opeR Bristol Robotics Lab\0	Operator
resP A Winfield +44 117 328 6913\0	person responsible
ebbN PyEBB v1.2\0	this EBB
chkS AF5679FC	checksum for this record

Table 4. An example Meta Data Record

The EBB Data Data Record The Data Data Record shall store information about the robot data records stored in the EBB.

Table 3 defines each field in the DD record, and includes both required and optional fields.

Notes on the DD fields:

1. For notes on *recS ebD1*, *ebT1*, *ebDM*, *ebTM* and *chkS* see notes on Table 3 above.
2. Field *ebbX* is an offset, in number of bytes, from the start of the EBB storage media to the next writable position for an RD record. Note that this will need to be reset back to RD 1 once the storage media is full.
3. Field *sysX* 'system exclusive' is manufacturer/operator definable. The data is formatted as 2 chars and a string separated by a colon, i.e. 00:string. The 2 characters are to allow the manufacturer to define up to 99 sysX fields, and the string allows for variable length data.
4. Given that all fields in the DD record are required and have a fixed length the *recS* field will have the default value 010:00000130, as shown in the example

label	data	length	requirement
recS	record size, field and chars	12 chars	required
ebbN	total number of EBB Data Records stored in EBB	10 chars	required
ebbX	index to the start of next writable RD record	16 chars	required
ebD1	date of oldest RD record written	10 chars	required
ebT1	time of oldest RD record written	12 chars	required
ebDM	date of most recent RD record written	10 chars	required
ebTM	time of most recent RD record written	12 chars	required
sysX	manufacturer definable field	variable	optional
chkS	Checksum for complete record	8 hexadecimal chars	required

Table 5. Data Data Fields

DD record in Table 6. Only if the record includes sysX field(s) will chkS have a different value.

An example of a complete Data Data Record is shown in 6.

field	comment
DD	record label
recS 010:000000130	number of fields:chars in record
ebbD 2022:04:20	date 20 April 2022
ebbT 16:40:20:000	time 16:40 and 20.000 seconds
ebbN 0000000400	400 records in EBB
ebbX 00000000001545060	offset to next RD position in storage media
ebD1 2022:03:01	first RD date 1 March 2022
ebT1 08:00:30:000	first RD time 08:00 and 30.000 seconds
ebDM 2022:05:01	Most recent RD date 1 May 2022
ebTM 18:59:30:100	Most recent RD time 18:59 and 30.100 seconds
chkS FF5678AC	checksum for this record

Table 6. An example Data Data Record

EBB Robot Data Records The Robot Data Records store operational data from the robot

Table 7 defines each field in the RD record, and includes both required and optional fields.

Notes on the RD fields:

1. For notes on *recS*, and *chkS* see notes on Table 3 above. For notes on *sysX* see notes on Table 5.
2. Field *botT* is needed in case the robot's clock shows a different time to the EBB's clock. The format of *botT* is the same as *ebbT*: 10 ASCII chars, for hours, minutes, seconds and milliseconds 00:00:00:000.

label	data	length	requirement
botT	robot time	10 chars	required
actD	actuator no and demand value	12 chars 000:±0000.00	optional
actV	actuator no and actual value	12 chars 000:±0000.00	optional
batL	battery level	3 chars	optional
tchS	touch sensor no and value	6 chars 00:000	optional
irSe	infra red sensor no and value	6 chars 00:000	optional
lfSe	line following sensor no and value	6 chars 00:000	optional
gyrV	gyro no and value	20 chars 00:±0000:±0000:±0000	optional
accV	accelerometer no and value	20 chars 00:±0000:±0000:±0000	optional
tmpV	temperature sensor no and value	8 chars 00:±0000	optional
micI	microphone no and input	variable, 2 chars:8 chars:wav hex	optional
camF	camera no and frame grab	variable, 2 chars:8 chars:jpg hex	optional
txtC	text input command	variable, string	optional
txtR	text reply	variable, string	optional
decC	robot decision code and reason	variable, 4 chars 0000:string	optional
wifi	WiFi status and signal strength	4 chars 0:00	optional
sysX	manufacturer definable field	variable, 2 chars 00:string	optional
chkS	checksum for complete record	8 hexadecimal chars	required

Table 7. Robot Data Fields

3. Fields *actD* and *actV* each contain values for the actuator number and the actuator demand, or actual positions respectively. The data is formatted 000:±0000.00 thus allowing for a maximum of 999 actuators, and positive or negative values of up to ±9999.99.
4. Fields *tchS, irSe, lfSe* each contain values for sensor number and the sensor value. The data is formatted 00:000 thus allowing for up to 99 sensors, and sensor values between 0...999.
5. Field *micI* allows for storage of a wav audio clip captured by the robot's microphone(s). The data has three colon separated elements, 2 chars for the microphone number, 8 chars for the length of the wav clip, and the hexadecimal representation of the wav binary. This allows for up to 99 microphones and wav clips of up to 99,999,999 bytes.
6. Field *camF* allows for storage of a jpg still frame grabbed by the robot's camera(s). The data has three colon separated elements, 2 chars for the camera number, 8 chars for the length of the jpg clip, and the hexadecimal representation of the jpg binary. This allows for up to 99 cameras and jpg clips of up to 99,999,999 bytes.
7. Fields *txtC* and *txtR* allow for storage of text input commands to the robot, and text responses from the robot, respectively. The input command might be typed by the robot's user, or spoken and then converted from speech to text by the robot's speech recognition system. The output text might be displayed visually or spoken by the robot's speech synthesis system. The data elements for both *txtC* and *txtR* are null terminated strings.

8. Field *decC* allows for storage of the robot's internal decision of its next action (i.e. turn left, turn right, stop, speak an alert, etc), together (optionally) with a reason for that decision. The data is formatting as a 4 char decision code, followed by a string. This allows for up to 9999 decisions to be logged. The determination of which numeric value to use for each robot decision is outside the scope of this standard, and left to the robot's manufacturer or operator. If there is no reason the string shall be stored as an empty null string terminated by ASCII \0.
9. Field *wifi* allows for storage of the robot's WiFi connection status and signal strength. The colon separated data is formatted as 1 character connection status, connected (1) or not connected (0), and 2 characters for signal strength, from 00..99.

An example of a complete Robot Data Record is shown in Table 6, for a simple differential drive wheeled robot, with 8 IR sensors.

field	comment
RD	record label
recS 017:00000nnn	number of fields:chars in record
ebbD 2022:04:20	RD date
ebbT 16:40:20:000	Rd time
batL 255	battery level
actV 001:-175.54	left wheel angle
actV 002:102.09	right wheel angle
irSe 001:0.05	IR sensor 1
irSe 002:0.05	IR sensor 2
irSe 003:0.05	IR sensor 3
irSe 004:0.05	IR sensor 4
irSe 005:0.05	IR sensor 5
irSe 006:0.23	IR sensor 6
irSe 007:0.15	IR sensor 7
irSe 008:0.05	IR sensor 8
decC 0020:obstacle detected\0	turning left to avoid obstacle on right
wifi 1:255	WiFi connected, good signal
chkS CFA3569A	checksum for this record

Table 8. An example Robot Data Record

A.4.2 EBB Timing Not all EBB records need to be written with the same frequency. In general those RD records which capture actuator movements and short range sensor inputs will need to be written with the highest frequency; a default setting for these might be once every 2 seconds. RD records that capture camera images might be written with a lower frequency, especially if the robot moves slowly, say once every 10 seconds, or less. A third group of RD records are

those that capture aperiodic events, such as a user's commands and the robot's response (if there is one).

Table 9 illustrates the EBB timing, with three kinds of event logged: high frequency motor and sensor values captured once every 2 seconds (in RD 1, 2, 4, 6 etc), lower frequency camera frame grabs once every 10 seconds (in RD 3, RD 10 and RD 17), and the sporadic events of user commands to override the robot's autonomous operation, in RD 5 and RD 13. For clarity only the *ebbT* time fields are shown in full. Abbreviated *camF* and *txtC* fields are also shown in RD 3, 5, 10, 13 and 17.

label	fields
MD	recS... ebbD... ebbT 08:40:20:000 BotN ePuck BotM... Resp... chkS...
DD	recS... ebbD... ebbT 08:40:20:000 ebbD1... ebbT1... ebbN... ebDM... ebbTM... chkS...
RD 1	recS... ebbD... ebbT 08:40:22:000 botT... actV... batL... irSE... chkS...
RD 2	recS... ebbD... ebbT 08:40:24:000 botT... actV... batL... irSE... chkS...
RD 3	recS... ebbD... ebbT 08:40:25:000 botT... camF 01:00307200:... chkS...
RD 4	recS... ebbD... ebbT 08:40:26:000 botT... actV... batL... irSE... chkS...
RD 5	recS... ebbD... ebbT 08:40:27:100 botT... txtC Halt\0 chkS...
RD 6	recS... ebbD... ebbT 08:40:28:000 botT... actV... batL... irSE... chkS...
RD 7	recS... ebbD... ebbT 08:40:30:000 botT... actV... batL... irSE... chkS...
RD 8	recS... ebbD... ebbT 08:40:32:000 botT... actV... batL... irSE... chkS...
RD 9	recS... ebbD... ebbT 08:40:34:000 botT... actV... batL... irSE... chkS...
RD 10	recS... ebbD... ebbT 08:40:35:000 botT... camF 01:00307200:... chkS...
RD 11	recS... ebbD... ebbT 08:40:36:000 botT... actV... batL... irSE... chkS...
RD 12	recS... ebbD... ebbT 08:40:38:000 botT... actV... batL... irSE... chkS...
RD 13	recS... ebbD... ebbT 08:40:27:100 botT... txtC Run\0 chkS...
RD 14	recS... ebbD... ebbT 08:40:40:000 botT... actV... batL... irSE... chkS...
RD 15	recS... ebbD... ebbT 08:40:42:000 botT... actV... batL... irSE... chkS...
RD 16	recS... ebbD... ebbT 08:40:44:000 botT... actV... batL... irSE... chkS...
RD 17	recS... ebbD... ebbT 08:40:45:000 botT... camF 01:00307200:... chkS...

Table 9. Example EBB illustrating variable frequency of RD sampling