SEDAP-Express
Interface Control Document (ICD)

I. Scope

SEDAP-Express is an exceptionally fast path to integrate new applications, sensors, effectors or other similar things into the ecosystem of MESE. That's why it is intentionally kept simple and offers several technical ways of communication. Of course, this results in limitations, but in most cases where quick and easy integration is required, these are negligible. If increased demands arise later on, the "bigger" SEDAP API respective MESE interface can be used if necessary. SEDAP-Express is licensed under the "Simplified BSD License" (BSD-2-Clause). Therefore, there should be no problems using SEDAP-Express in commercial or non-commercial projects or integrating parts of the SEDAP-Express framework.
Everything you need for development and testing can be found on the Internet at http://SEDAP.Express.

II. Glossary

MESE	=	Military Expandable Software Environment
SEDAP	=	Safety critical Environment for Data exchange And Process scheduling
CSV	=	Comma-Separated-Values
SEC	=	SEDAP-Express-Connector
SECMockUp	=	Simulation of the real SEC
C2MockUp	=	Simulation of a C2 system with a simple map
MessageTool	=	Tool for manual generation of messages
SIDC	=	Symbol identification code (APP-6A/B/MIL-STD-2525B/C/STANAG 2019)
ASCII	=	American Standard Code for Information Interchange – in this context the ISO-8859-1 table is meant
Base64	=	Binary-to-text encoding scheme, which is using an alphabet of 64 characters

III. General connection attributes

1. Common conventions

- Basic format is CSV using ; (0x3B) as separation character, with a \n (0x10) as end termination
- Elements of lists are separated by # (0x23), Mandatory fields/elements except the name are marked with (M)
- The messages are human-readable and using the ASCII-table
- (Binary)Data which possibly contains a special character (0x10, 0x23, 0x3B) has to be encoded with Base64
- Unknown/Invalid values must not be transmitted, the respective field will be left empty
- If there are only ; characters left in the message, these could be cut off
- Support for IPv4 or IPv6 (except for serial connection)
- SEC/SECMockUp/Applications can send and receive at any time
- Application shall send heartbeat message not more often than with 1Hz (+-100ms), but can vary if it is required
- SEC/SECMockUp answers heartbeat also with a heartbeat message (see chapter IV.2.12)

2. Authentification

- If authentication/encryption is required, it's always preferred to use VPN if available
- Messages could be authenticated by fulfilling the HMAC field using a password (see chapter IV.1.1)
- The password can either be defined in advance or exchanged using the Diffie-Hellman process and the KEYEXCHANGE message (see chapter IV.2.13). It's recommended to authenticate even this message with a pre-shared password.
- For calculating the HMAC you have to use all message fields (see chapter IV.1.1.1) and setting temporary hmac "0000"
- At minimum standards defined by FIPS 140-2 (Federal Information Processing Standard) have to be used
  That means it is preferred to use:
  - 32Bit/64Bit CMAC in combination AES128 (NIST SP 800-38B)
  - 32Bit/64Bit GMAC in combination AES128 (NIST SP 800-38D)
- It's recommended to use HMAC DBRG (Deterministic Random Bit Generator, NIST SP 800-90A/B)

  Sample (32Bit CMAC, Password:expressexpressex):
  OWNUNIT;5E;661D4410;66A3;R;;;53.32;8.11;0;5.5;21;22;;;FGS Bayern;sfspfclff------

3. Encryption

- Encryption is optional
- At minimum standards defined by FIPS 140-2 (Federal Information Processing Standard) have to be used
  That means it is preferred to use:
  - AES128/256 ECB (NIST SP 800-38A)
  - AES128/256 CBC (NIST SP 800-38A)
  - XOR (Pseudo-encryption ONLY for testing/debugging purposes or for very light obfuscation)
- It's recommended to use HMAC DBRG (Deterministic Random Bit Generator, NIST SP 800-90A/B)
- Encrypted data must be Base64 encoded - all incl. header (see chapter IV.1.1.1) have to be encoded
- If there is password given every message have to be encrypted – mixture of encrypted and plain message are not allowed

Sample reference message:
OWNUNIT;5E;661D4410;66A3;R;;;53.32;8.11;0;5.5;21;22;;;FGS Bayern;sfspfclff------

Sample encrypted message (AES128, ECB, Password:expressexpressex):
SpkxMb4T08Py6MDfwRJUylJLE45edpyrZ3pFSw0vWdbk/Ry1RKeSx1gFCpzGhVLsfx0iNQ6fuUwtG9UfweXRSvN5Lk0XMN6TYAc4TOH
os0I=

4. Compression

- Compression is optional (has to be used BEFORE an optional encryption because of the high entropy)
- Use the widely spreaded "deflate" for compression (only effective if the message size exceeds 140 characters)
- Compressed data must be Base64 encoded – all incl. header (see chapter IV.1.1.1) have to be compressed
- If the first bytes of a received message doesn't match a message name, prove for compression

Sample reference message:
TEXT;czG1NjMzdDEzNTO1NjYycbUOtrY2slYKycgsVgCiRIW8/JLM5FRFJQA=

Sample compressed message:
eJwLcY0IsXYxszYzM3QxNzA2sjY2MnG1Dra2NrZWCsnILFYAokSF5IzEEoXc1OLixPRURSVrVxM3QwC2Vg/Y

5. TCP Connection

- Standard port 50000, but customizable
- SEC/SECMockUp = Server (1)
- Application = Client (n)

6. UDP-Connection

- Standard port 50000, but customizable
- Support for Uni-, Broad- or Multicast mode
- Standard Multicast-Address is 228.2.19.80 resp. ff02:8:2:19:80::1
- Multiple messages per UDP-packet possible
- SEC/SECMockUp answers heartbeat message with UDP-Unicast (see chapter IV.2.12)

7. Serial connection

- Standard 115200-8-N-1, Full-Duplex is preferred, Half-Duplex and Simplex without acknowledge requests
- There are three modes available, depending on the use case:
    - Message never contains a \n (0x10), therefore the message could be sent with an additional appended \n\n\n\n
    - Message contains one or more \n (0x10), have to be Base64-encoded and sent with an additional appended \n\n\n\n

8. REST-API

- Standard ports are HTTP 80 or HTTPS 443, but customizable
- Deflate or gzip compression should be supported

9. Protocol Buffers

- Ports and other TCP or UDP parameters are the same as descripted in chapter III.5 and III.6.

IV. Data Exchange between SEC/SECMockUp and client applications

1. General

1.1. TCP-/UDP-/Serial connections

On principle, messages have a CSV structure with a common header:
<Name>(M);<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;<Content>

Everything except the name is basically optional. Certain messages require specific header elements, such as time or the sender. These exceptions are described in the chapter of the respective message. Hexadecimal numbers have no 0x-prefix.

<Name>	Defines the purpose of a message. Sometimes it's so-called topic.
<Number>	This is a hexadecimal string representation of an 8-bit sequential number. Each type of message has its own counter that starts again with zero after reaching 255/FF. A reconnect resets the counters!
<Time>	A hexadecimal string representation of a 64-bit Unix time stamp with milliseconds.
<Sender>	This field specify the original sender of the message information. In most cases one should use a hexadecimal string representation of a 16-bit unsigned integer but one can also use free text. This field won't be changed, even if a message has been forwarded or relayed. This sender identification can be chosen randomly by the participants themselves or permanently assigned by a responsible institution when preparing a specific use/network. If information of a sub-system has to be forwarded the sender identification should be the source of the original information, in that case of the sub-system.
<Classification>	Describes the classification or security level of the content. Possible values are P=public, U=unclassified, R=restricted, C=confidential, S=secret, T=top secret
<Acknowledgement>	TRUE=request an acknowledgement, FALSE/Nothing=No acknowledgement
<HMAC>	Hash-based message authentication code for verification
<Content>	Content of the message, depending on the message purpose.

1.2. REST-API connection

If the REST-API shall be used, it's preferred to also use the provided JSON schema file and the generated code which either comes with the SEDAP-Express SDK or has been generated by yourself. You can find the schema in chapter IV.3 or on http://sepap.express.

1.3. Protobuf connection

At least it's also possible to use Google™ protocol buffers to exchange the SEDAP-Express messages. You can find the schema in chapter IV.4 or on http://sepap.express. This allows you to generate your own code or use the existing code from the framework or the sample client.

2. Messages

This is the list of all so-far available messages and their structures and content including some samples. All units of measurement are generally given in SI-units, but there are deviations where this makes sense due to the usual range of values. In the following the used units will be given within square brackets for all message-descriptions. The altitude is the altitude above sea-level. A value of zero means exactly on ground, if the position is on land. Latitude and longitude are in decimal degrees, while positive values means north and east respective negative values south and west. Relative position values are defined this way, that the x-axis points to the west direction, y-axis points to the north and the z-axis is equal to the height above the unit. Speed and course are meant to be relative to ground. Course and heading have a range from zero to 359,999 and are relative to geographic north or zero degree. In general, all values are mandatory. Optional parameters are marked with .

2.1. OWNUNIT

Description: Positional, kinematic and identification data of the own (sent by the client) or host (sent by the SEC) unit/platform. If a client is sending this message, it will be converted to a contact and sent into the MESE network.

Structure: OWNUNIT;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;
<Latitude>[°](M);<Longitude>[°](M);<Altitude>[m];<Speed over ground>[m/s];<Course over ground>[°];
<Heading>[°];<Roll>[°];<Pitch>[°];<Name>;<SIDC>

Sample 1: OWNUNIT;5E;661D4410;66A3;R;;;53.32;8.11;0;5.5;21;22;;;FGS Bayern;sfspfclff------
Sample 2: OWNUNIT;5E;661D4410;66A3;R;TRUE;;42.32;-123.11;10000;50.23;297;;;33.3;-0.15;sfapmf---------

2.2. CONTACT

Description: Positional, kinematic and identification data of a contact. For example, this message would be used by a sensor to report a contact it recognized. In return this message would be used to receive the tactical picture from the MESE network.

Structure:
CONTACT;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;<ContactID>(M);<DeleteFlag>;
<Latitude>[°](M);<Longitude>[°](M);<Altitude>[m];<relative X-Distance>[m];<rel Y-Distance>[m];<rel Z-Distance>[m];
<Speed over ground>[m/s];<Course over ground>[°];<Heading>[°];<Roll>[°];<Pitch>[°];
<width>[m];<length>[m];<height>[m];<Name>;<Source>;<SIDC>;<MMSI>;<ICAO>;<Image>;<Comment>

ContactID	ASCII	A positive identification unique number or free text of the contact chosen by the sender of this message
DeleteFlag	TRUE	Contact has to be removed
	FALSE	Contact is current
Source	ASCII	Available types (more than one available): R=Radar,A=AIS,I=IFF/ADS-B,S=Sonar,E=EW,O=Optical,Y=Synthetic,M=Manual
SIDC	SIDC	Identification code
MMSI	MMSI	Maritime Mobile Service Identity
ICAO	ICAO	International Civil Aviation Organization
Image	Base64	Imagedata (JPG, PNG, TIF) encoded in Base64
Comment	ASCII	Free text to the contact

Sample 1: CONTACT;5E;661D4410;66A3;R;;;100;FALSE;53.32;8.11;0;5.5;21;22;;;;;;FGS Bayern;AR;sfspfclff------;;;;Ch22
Sample 2: CONTACT;5F;661D5420;66A3;U;;;101;FALSE;36.32;12.11;2000;44;331;;11;65;10;Unknown;O;;22113321;;;NL
Sample 3: CONTACT;60;661B7410;66A3;S;TRUE;;102;TRUE;53.32;8.11

2.3. EMISSION

Description: Positional, attributes and identification data of an electro-magnetic, optical or acoustic emission.

Structure:
EMISSON;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;
<EmissionID>(M);<DeleteFlag>;<SensorLatitude>[°](M);<SensorLongitude>[°](M);<SensorAltitude>[m];
<EmitterLatitude>[°];<EmitterLongitude>[°];<EmitterAltitude>[m];<Frequency[Hz]>;<Bandwidth[Hz]>;
<Power[db(A)]>;<FreqAgility>;<PRFAgility>;<Function>;<SpotNumber>;<SIDC>;<Comment>

EmissionID	Number>0	A positive identification unique number of the emission chosen by the sender of this message. This number should also be unique in terms of contact numbers.
DeleteFlag	TRUE	Emission has to be removed
	FALSE	Emission is current
FreqAgility	0	Stable_Fixed
	1	Agile
	2	Periodic
	3	Hopper
	4	Batch hopper
	5	Unknown
PRFAgility	0	Fixed periodic
	1	Staggered
	2	Jittered
	3	Wobbulated
	4	Sliding
	5	Dwell switch
	6	UnknownPRF
	7	CW
Function	0	Unknown
	1	Esm_Beacon/Transponder
	2	Esm_Navigation
	3	Esm_Voice_Communication

		4	Esm_Data_Communication
		5	Esm_Radar
		6	Esm_Iff
		7	Esm_Guidance
		8	Esm_Weapon
		9	Esm_Jammer
		10	Esm_Natural
		11	Acoustic_Object
		12	Acoustic_Submarine
		13	Acoustic_Variable_Depth_Sonar
		14	Acoustic_Array_Sonar
		15	Acoustic_Active_Sonar
		16	Acoustic_Torpedo_Sonar
		17	Acoustic_Buoys_Sonar
		18	Acoustic_Decoy_Signal
		19	Acoustic_Hit_Noise
		20	Acoustic_Propeller_Noise
		21	Acoustic_Underwater_Telephone
		22	Acoustic_Communication
		23	Acoustic_Noise
		24	Laser_Range_Finder
		25	Laser_Designator
		26	Laser_Beam_Rider
		27	Laser_Dazzler
		28	Laser_Lidar
	SIDC	SIDC	Identification code
	Comment	ASCII	Free text to the emission

Sample 1: EMISSION;5E;661D4410;66A3;R;;;100;;53.32;8.11;0;54.51;8.15;0;8725000;20000;3;0;2;6;5;10233;SA-8
Sample 2: EMISSION;5F;661D5410;66A3;R;;;101;;54.86;9.32;0;52.12;9.80;50;2572500;40000;1,5;2;0;6;;sngpesr--------

2.4. METEO

Description: Metrological data of the environment.

Structure:
METEO;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>; <HMAC>;
<SpeedThroughWater>[m/s];<WaterSpeed>[m/s];<WaterDirection>[°];<WaterTemperature>[°C];<WaterDepth>[m];
<AirTemperature>[°C];<DewPoint>[°C];<HumidityRel>[%];<Pressure>[hPa];<WindSpeed>[m/s];<WindDirection>[°];
<Visibility>[km];<CloudHeight>[m];<CloudCover>[%]

Sample: METEO;AC;661D44C0;74BE;U;;;15.4;15.5;;;;10.2;72;20.3;;55;1005;25;;;2500;33

2.5. TEXT

Description: Human readable textual data. This could be an alert message, but also a simple text message for chatting.

Structure: TEXT;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;
<Type>;<Text>(M);<Recipient>

Type	0	Alert
	1	Warning
	2	Notice
	3	Chat
Text	ASCII	Free text of the message
Recipient	HexString	This field specifies the recipient for the message by hexadecimal string representation of a 16-bit unsigned integer, as explained in table form chapter IV.1.1

Sample 1: TEXT;D3;661D44D2;324E;S;TRUE;;0;"This is an alert!"
Sample 2: TEXT;D4;661D458E;324E;S;TRUE;;1;"This is a warning!"
Sample 3: TEXT;D5;661D6565;324E;S;;;2;"This is a notice!"
Sample 4: TEXT;D6;661D7032;324E;S;;;3;"This is a chat message!";E4F1

2.6. COMMAND

Description: Command for one specific or all possible recipients. Which camera is assigned to which number and what camera modes are available, have to be defined specifically for every use case depending on the sensor platform. The same is applies for the kind of action.

Structure: COMMAND;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<Recipient>(M);<CmdType>(M);<additional cmd-dependent parameters>*

Recipient	HexString	This field specify the recipient of the command by hexadecimal string representation of a 16-bit unsigned integer.
CmdType	0	Power off device: <Unix time stamp>
	1	Restart device: <Unix time stamp>
	2	Set device into standby: <Unix time stamp>
	3	Wake up device
	4	Sync time: <IP/Hostname of a NTP server>
	5	Send status
	6	Move: <Latitude>[°];<Longitude>[°];<Altitude>[m]
	7	Scan Area: <Latitude1>[°];<Longitude1>[°];<Latitude2>[°];<Longitude2>[°];<RotationAngle>
	8	Action: <Kind of action>
	9	Take photo: <Number of camera>;<Camera mode>
	10	Switch on video stream: <Number of camera>;<Camera mode>
	11	Switch off video stream: <Number of camera>
	12	Start engagement: <contactID>
	13	Stop engagement: <contactID>

Sample 1: COMMAND;27;661D44C0;E4B3;C;TRUE;;AB49;2
Sample 2: COMMAND;28;661D44C0;E4B3;C;TRUE;;AB49;12;1000
Sample 3: COMMAND;29;661D44C0;E4B3;C;TRUE;;4;10.0.0.1

2.7. GRAPHIC

Description: Define graphical plans likes polygons, squares or routes

Structure:
GRAPHIC;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;<GraphicType>(M);
<LineWidth>;<LineColor>;<Annotation>;<additional GraphicType-dependent parameters>*

GraphicType	0	Point: <Latitude>[°];<Longitude>[°];<Altitude>[m]
	1	Path: <Latitude>[°],<Longitude>[°],<Altitude>[m] # …
	3	Polygon: <Latitude>[°],<Longitude>[°],<Altitude>[m] # …
	4	Rectangle: <RotationAngle>[°];<Latitude1>[°],<Longitude1>[°],<Altitude1>[m]#<Latitude2>[°], <Longitude2>[°],<Altitude2>[m]
	5	Square: (t.b.d.)
	6	Parallelogram: (t.b.d.)
	7	Trapezium: (t.b.d.)
	8	Circle: <radius>[m];<Latitude>[°];<Longitude>[°];<Altitude>[m]
	9	Ellipse: <radius-X>[m];<radius-Y>[m]; <CenterLatitude>[°];<CenterLongitude>[°];<CenterAltitude>[m]
	10	Block: (t.b.d.)
	11	Sphere: (t.b.d.)
	12	Cone: (t.b.d.)
	13	Pyramid: (t.b.d.)
	14	Ellipsoid: <X-Radius>[m];<Y-Radius>[m];<Z-Radius>[m]; <Center_Latitude>[°];<Center_Longitude>[°];<Center_Altitude>[m]
LineWidth	=> 1	Width of the line or the point
LineColor	RGB	Color of the line or the point in Web notation 800000 for a darker red
Annotation	ASCII	Text for an annotation to this graphic

Sample 1: GRAPHIC;77;661D64C0;910E;U;;;0;1;FF0000;StartPoint;54.23;12.86
Sample 2: GRAPHIC;78;661D64C0;910E;U;;;1;1;808080;Transit;54.23,12.86#54.30,12.9#54.55,13.3
Sample 3: GRAPHIC;79;661D62C0;910E;U;;;8;1;FF8000;Area A;10000;53.43;9.45

2.8. STATUS

Description: This message offers the possibility to check the connection, which is primarily important, if you are using UDP or serial connection. It should not be sent more often than 1Hz. Nevertheless, if it is needed – one can use a faster repetition. The receiver field is optional and can be one single recipient or a list of recipients. As also descripted in the header definition (chapter IV.1.1.1), if information of a sub-system has to be forwarded the sender identification should be the source of the original information. For instance, if there is a swarm of drones, one should use the concrete drone id as sender identification.

Structure:
STATUS;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<TecStatus>;<OpsStatus>;
<FuelLevel>;<BatterieLevel>;<FreeText>

TecStatus	0	Not operational
	1	Initializing
	2	Degraded
	3	Partly operational
	4	Fully operational
	5	Fault
OpsStatus	0	Not operational
	1	Initializing
	2	Degraded
	3	Partly operational
	4	Fully operational
AmmunitionLevel	%	Relative remaining ammunition
FuelLevel	%	Relative remaining fuel capacity
BatterieLevel	%	Relative remaining batterie capacity
IP/Hostname	ASCII	IP or hostname of the platform
Media	Base64	List of video stream or image URLs
FreeText	ASCII	Human readable free text description of the status

Sample 1: STATUS;15;661D44C0;75DA;U;;;4;2;20;;50;;Fully operational
Sample 2: STATUS;16;661D64C0;129E;R;;;2;2;10;;;aHR0cDovLzEwLjAuMC4xL2ltYWdlLnBuZw==;Out of fuel!
Sample 3: STATUS;17;661D64C0;129E;R;;;4;4;;;50;cnRzcDovLzEwLjAuMC4xMC9zYW1wbGVTdHJlYW0=

2.9. ACKNOWLEDGE

Description: If a client or the SEC requested an acknowledge of a packet one has to use this this message. The acknowledgement flag is fixed set to FALSE. The awaiting client or SEC have to wait maximal 2 seconds before resending the original message with set acknowledgement flag.

Structure: ACKNOWLEDGE;<Number>;<Time>;<Sender>;<Classification>;;<HMAC>;
<Receiver>(M);<Name>(M);<Counter>(M)

Recipient	HexString	This field specify the recipient of the acknowledge by hexadecimal string representation of a 16-bit unsigned integer
Name	ASCII	The name of the message which should be acknowledged
Counter	Number	The number of the message which should be acknowledged. This is a hexadecimal string representation of an 8-bit.

Sample: ACKNOWLEDGE;18;661D64C0;129E;R;;;FE2A;COMMAND;31

2.10. RESEND

Description: Missing messages can be requested again with this message. These messages can be recognized by the message number in the header, the sender ID and the message name as described in chapter IV.1.1.1.

Structure: RESEND;<Number>;<Time>;<Sender>;<Classification>;;<HMAC>;
<Receiver>(M);<Name>(M);<name of the missing message>(M);<number of the missing message>(M)

Sample: RESEND;20;661D64C0;129E;R;;;FE2A;TEXT;31

2.11. GENERIC

Description: This message is an empty container for transporting any kind of data. It has to be defined in the respective case. For example, one can use it to exchange the original MESE/SEDAP messages or other propriety protocol data. In the last case you have to use any other self-defined type.

Structure: GENERIC;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;
<HMAC>;<ContentType>;<EncodingFlag>;<Content>

ContentType	SEDAP	Content is an original MESE message
	*	Self-defined ASCII string
EncodingType	TRUE	Content is Base64 encoded
	FALSE	Content is NOT encoded
Content		Any content in printable ASCII or Base64 encoded

Sample 1: GENERIC;5E;661D4410;66A3;R;;;SEDAP;FALSE;
Sample 2: GENERIC;5E;661D4410;66A3;R;TRUE;;SEDAP;TRUE;U2FtcGxlIGJpbmFyeSBkYXRhIEdyZWV0aW5ncyA6RA==
Sample 3: GENERIC;5E;661D4410;66A3;R;;;RADNMEA;;$RATTM,11,11.4,13.6,T,7.0,20.0,T,0.0,0.0,N,,Q,,154125.82,A,*17

2.12. HEARTBEAT

Description: This message offers the possibility to check the connection, which is primarily important, if you are using UDP or serial connection. It should not be sent more often than 1Hz. Nevertheless, if it is needed – one can use a faster repetition. The receiver field is optional and can be one single recipient or a list of more than one recipient. If no recipient is provided than all possible receivers in the network/serial net are addressed. A heartbeat message has an empty acknowledgement flag, cause you cannot request one for it. Besides this, the acknowledgement flag is fixed set to FALSE (empty field).

Structure:
HEARTBEAT;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;<Recipient>

Recipient	HexString	This field specify the recipient of the command by hexadecimal string representation of a 16-bit unsigned integer.

Sample 1: HEARTBEAT;42;661D5420;89AD;U;;;FE2A
Sample 2: HEARTBEAT;43;;1022
Sample 3: HEARTBEAT;43;
Sample 4: HEARTBEAT

2.13. KEYEXCHANGE

Description: If you don't have the possibility to exchange a password/key on another channel (e.g. mail, telco), this message can be used to exchange keys via Diffie-Hellman-Merkle process. It's preferred to use ECDH or standard DH with HMAC DRBG.
If possible, also use HMAC authentication for these messages or otherwise do some plausibility checks.

Structure:
KEYEXCHANGE;<Number>;<Time>;<Sender>;<Classification>;<Acknowledgement>;<HMAC>;
<Receiver>(M);<Phase>(M);<Key length>(M);<Prime>(M);<Natural Number>(M);<Public key>(M)

Algorithm	0	DH (Diffie-Hellman-Merkle, NIST SP 800-56A/B, NIST SP 800-90A/B)
	1	ECDH (Diffie-Hellman-Merkle with Curve25519 / X25519, RFC 7748)
Phase	0	Exchange the public variables (DH only)
	1	Exchange public keys
Key length	128/256	Bit Length of the key (Phase 0)
Prime (p)	HexString	Publicly known prime number (> 3000 bits / 375 byte) (Phase 0, DH only)
Natural number (g)	HexString	Publicly known natural number smaller than p (Phase 0, DH only)
Public key	HexString	The public key of the sender (Phase 1)

Sample 1: KEYEXCHANGE;0;661D5420;89AD;U;;;FE2A;0;128;7FFFFFFF;822460DE
Sample 2: KEYEXCHANGE;0;661D5430; FE2A;U;;;89AD;1;128;;;6E6026EFF9D9EBEB9D4A973CB5C287DBD77D75EDDD2

3. SEDAP-Express JSON-Schema

The JSON schema was intentionally kept very simple. The JSON message contains only a list of one or more SEDAP-Express messages in their original (JSON-compatible) format. This means that the same message classes could be used for parsing and generating.

Schema:

```
{
  "messages":[
    {
      "message":""
    }
  ]
}
```

Sample:

```
{
  "messages":[
    {
      "message":"CONTACT;60;661B7410;66A3;S;TRUE;102;TRUE;53.32;8.11"
      "message":"METEO;AC;661D44C0;74BE;U;;15.4;15.5;;;10.2;72;20.3;;55;1005;25;;;2500;33"
      "message":"TEXT;D6;661D7032;324E;S;;3;"This is a chat message!";E4F1"
      "message":"GRAPHIC;79;661D62C0;910E;U;;8;1;FF8000;Area A;10000;53.43;9.45"
    }
  ]
}
```

4. SEDAP-Express Protobuf-Definition

The Google™ Protocol buffer definition is kept very simple, too. As with JSON messages, the same classes could be used to parse or generate the actual content.

Definiton:

```
syntax = "proto3";

message SomeMessage {

   message Messages {
      string message = 1;
   }

   repeated Messages messages = 1;
}
```

V. Contact

Federal Armed Forces of Germany
Naval Support Command II A
c/o Volker Voß
Wibbelhofstraße 3
26384 Wilhelmshaven
Germany

E-Mail: mese@bundeswehr.org
Tel.: +49 4421 68 67290